

AD-A241 376



NAVAL POSTGRADUATE SCHOOL  
Monterey, California

2



DTIC  
ELECTE  
OCT 10 1991  
S B D

THESIS

THE PATH PREDICTION OF CYCLONES WITH KALMAN  
FILTERS

by

Dogan Taskin

September 1990

Thesis Advisor:

Prof. Harold Titus

Approved for public release; distribution is unlimited

91-12668



91 12668 011

REPORT DOCUMENTATION PAGE

Form Approved  
OMB No. 0704-0188

1a REPORT SECURITY CLASSIFICATION <b>UNCLASSIFIED</b>		1b RESTRICTIVE MARKINGS	
2a SECURITY CLASSIFICATION AUTHORITY		3 DISTRIBUTION AVAILABILITY OF REPORT Approved for public release; distribution is unlimited	
2b DECLASSIFICATION/DOWNGRADING SCHEDULE			
4 PERFORMING ORGANIZATION REPORT NUMBER(S)		5 MONITORING ORGANIZATION REPORT NUMBER(S)	
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b OFFICE SYMBOL (If applicable) EC	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
6c ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000		7b ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	
8a NAME OF FUNDING/SPONSORING ORGANIZATION	8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8c ADDRESS (City, State, and ZIP Code)		10 SOURCE OF FUNDING NUMBERS	
		PROGRAM ELEMENT NO	PROJECT NO
		TASK NO	WORK UNIT ACCESSION NO
11 TITLE (Include Security Classification) The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the US Government. <i>See Cover</i>			
12 PERSONAL AUTHOR(S) TASKIN, Dogan			
13a TYPE OF REPORT Master's Thesis	13b TIME COVERED FROM _____ TO _____	14 DATE OF REPORT (Year, Month, Day) 1990 September	15 PAGE COUNT 73
16 SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
17 COSAT CODES		18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUBGROUP	
		Kalman Filter; path prediction of cyclones; Turkey	
19 ABSTRACT (Continue on reverse if necessary and identify by block number)			
<p>The Kalman filter is used to provide estimates of the position and velocity of a storm based upon observation of the storm's longitude and latitude. Nonstationary noise is shown to degrade the performance of the filter and cause tracking divergence. Time varying values for the noise covariance matrices R and Q, and the addition of an external forcing function to the filter, effectively compensated for this tracking error.</p> <p>Results for the simulations show significant performance advantages of using an external forcing function in the system.</p>			
20 DISTRIBUTION AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED, LIMITED <input type="checkbox"/> SAME AS REF <input type="checkbox"/> DTIC USERS		21 ABSTRACT SECURITY CLASSIFICATION <b>UNCLASSIFIED</b>	
22a NAME OF RESPONSIBLE INDIVIDUAL TITUS, Hal		22b TELEPHONE (include Area Code) 408-646-2560	22c OFFICE SYMBOL EC/Ts

Approved for public release; distribution is unlimited.

The Path Prediction of Cyclones with Kalman Filters

by

Dogan Taskin  
Lieutenant Junior Grade, Turkish Navy  
B.S., Turkish Naval Academy, 1984

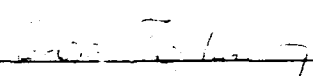
Submitted in partial fulfillment  
of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

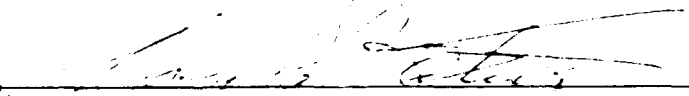
from the

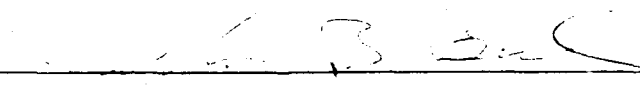
NAVAL POSTGRADUATE SCHOOL  
September 1990


Author:

  
Dogan Taskin

Approved by:

  
Harold A. Titus, Thesis Advisor

  
Jeffrey B. Burl, Second Reader

  
Michael A. Morgan, Chairman  
Department of Electrical and Computer Engineering

# ABSTRACT

The Kalman filter is used to provide estimates of the position and velocity of a storm based upon observation of the storm's longitude and latitude. Nonstationary noise is shown to degrade the performance of the filter and cause tracking divergence. Time-varying values for the noise covariance matrices R and Q, and the addition of an external forcing function to the filter effectively compensated for this tracking error.

Results for the simulations show significant performance advantages in using external forcing functions in the system.



iii

Accession For	
NTIS CEASI	<input checked="" type="checkbox"/>
DETC T&E	<input type="checkbox"/>
Unpublished	<input type="checkbox"/>
Justification	
By _____	
Dist _____	
Availability Codes	
Dist	Special
A-1	

## TABLE OF CONTENTS

I. INTRODUCTION . . . . .	1
A. GENERAL . . . . .	1
B. OBJECTIVES OF THIS THESIS . . . . .	1
C. THESIS ORGANIZATION . . . . .	2
II. KALMAN FILTER . . . . .	3
A. GENERAL . . . . .	3
B. THEORETICAL BASIS OF THE KALMAN FILTER . . . . .	3
C. KALMAN FILTER EQUATIONS . . . . .	6
1. Kalman Gain . . . . .	7
2. Error Covariance . . . . .	7
3. Residual . . . . .	8
4. Extended Kalman Filter . . . . .	10
III. PROBLEM STATEMENT . . . . .	12
A. PHYSICAL SYSTEM . . . . .	12
B. STATE SPACE MODEL . . . . .	13
1. Prediction of Cyclone Paths . . . . .	15
2. State Excitation Covariance Matrix Q . . . . .	20
C. MEASUREMENT MODEL . . . . .	23
1. The Measurement Noise Covariance Matrix R . . . . .	23

D. SMOOTHING ALGORITHM . . . . .	25
IV. COMPUTER SIMULATIONS . . . . .	27
A. GENERAL . . . . .	27
1. Typhoon Hal . . . . .	28
2. Typhoon Doyle . . . . .	35
3. Typhoon Uleki . . . . .	42
V. CONCLUSION . . . . .	49
APPENDIX STORM.FOR SOURCE CODE . . . . .	51
LIST OF REFERENCES . . . . .	71
BIBLIOGRAPHY . . . . .	72
INITIAL DISTRIBUTION LIST . . . . .	73

### ACKNOWLEDGMENT

I would like to thank my advisor, Hal Titus. His loyal guidance blended with morale-boosting humor fostered a professional and productive work environment. I would also like to thank my friends who supported me. Special thanks go to Capt. Steve and Lt. Dan, whose support was never ending.

## **I. INTRODUCTION**

### **A. GENERAL**

The western North Pacific Ocean is the most active tropical cyclone basin in the world. The need for accurate storm forecasts is of utmost importance to the civilian and military communities. The loss of both life and property from these storms can be considerable. The early sailors recognized that the low pressures were associated with high winds rotating counterclockwise around the center in the Northern Hemisphere and clockwise in the Southern Hemisphere. They also knew of the dangerous winds and heavy weather to the right of the center in the Northern Hemisphere and to the left in the Southern Hemisphere, and adjusted their sailing practices accordingly to minimize the amount of damage caused by these severe storms and to provide more accurate warnings to ships and shore facilities. This study develops a tropical cyclone track prediction model using a Kalman Filter with smoothing.

### **B. OBJECTIVES OF THIS THESIS**

This thesis will be an extension of a previous thesis done by LTJG Asim Mutaft [Ref.2]. The major points of that thesis were

- Development of a Kalman Filter storm tracking program
- Fictitious noise source for state excitation matrix  $Q_k$



- The position errors achieved by this program were 10-15 nautical miles

This work attempts to improve on the previous research by implementing deterministic forcing functions and a maneuver divergence detection scheme that uses a noise variance estimator process. This research investigates the behavior of a Kalman Filter in tracking a storm by means of latitude and longitude observations.

The estimation of the forcing function, directional and speed deviation is very important in a storm position estimate. By having a more accurate assessment of what the storm has done in the past, we will be better able to predict and estimate a storm's future course, speed and position.

### **C. THESIS ORGANIZATION**

Chapter I states the problem of concern and serves as an introduction to the report. Chapter II gives a mathematical derivation of the Kalman Filter equations and explanations and comments. Chapter III model's storm tracking and prediction. Chapter IV shows the simulations and gives results. Finally, Chapter V lists the conclusions. The appendics contains the program code.

## II. KALMAN FILTER

### A. GENERAL

The Kalman Filter has been in use since 1960 in the design of estimation systems. Kalman introduced the filter as a state space representation of a linear time invariant system. Modelling of this system has the advantage of maintaining the system's physical state in a matrix equation model.

The terms and parameters for the equations are listed in Table 2.1. Terms appearing with single subscripts refer to a value of the term at a given time while dual subscripts refer to the term's values at the time of the first subscript and containing observation data ending with the last.

### B. THEORETICAL BASIS OF THE KALMAN FILTER

There are several methods to model a system bilinear transformation, state space approximation, and pole zero mapping, etc. The method used here is state space approximation. The linear system's state space model is depicted in Eq. (2.1). The measurements taken during system parameter estimation are given in Eq. (2.2), where  $z_k$  represents observed parameters (bearing, range, etc.) and  $x_k$  represents the physical state of the system (position, velocity, etc.).

$$x_{k+1} = \phi x_k + w_k \quad (2.1)$$

$$z_{k+1} = Hx_{k+1} + v_{k+1} \quad (2.2)$$

These standard linear difference equations are time invariant as the equation matrices do not vary with the time subscripts.

TABLE 2.1 DEFINITION OF TERMS

Identity matrix	I
System state	$x_k$
State transition matrix	$\phi$
State excitation noise	$w_k$
Observation	$z_k$
Observation matrix	H
Observation noise	$v_k$
State estimate (Predicted)	$\hat{x}_{k+1/k}$
Estimate error	$\tilde{x}_{k+1/k}$
Expected value of the error	$E[\tilde{x}_{k+1/k}]$
Error covariance matrix	$P_{k+1/k}$
Residual	$r$
Kalman gain	G

The filtering process estimate of the state vector at the present time, depends on the present and past measurements. In order to this, the filter needs *a priori* information of the state estimate, its error covariance matrix, and the actual observation.

The matrix  $\phi$  is chosen to fit the target dynamics in Eq. (2.1). The target dynamics are usually expected to be stationary and moving linearly at constant velocity. The appropriate  $\phi$  matrix is represented in Eq. (2.3).

$$\phi = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.3)$$

while the constant,  $T$ , is the observation interval.

$H$  is the observation matrix. Calculating the observation matrix requires precise knowledge of the state of the system. When the system is Linear Time Invariant,  $H$  is constant.

The noise is assumed to be Additive White Gaussian Noise. This is an idealization, often justified in real systems. The statistical properties of the input and observation noise covariance matrices,  $Q$  and  $R$ , respectively are as followed.

$$E[w_k] = E[v_k] = 0 \quad (2.4)$$

$$E[w_k w_k^T] = Q \quad (2.5)$$

$$E[v_k v_k^T] = R \quad (2.6)$$

Correct estimation of the observation noise,  $v_k$ , is critical to the Kalman Filter's performance. Two approaches of estimating the noise and the effect on optimal filter performance will be examined.

Observations received by the Kalman Filter includes the unwanted noise plus the desired information. The desired behavior is to de-emphasize the noise and react to the information only. The ability to perfectly predict the states, requires the setting of the Kalman gains to maximizing the extracted information.

When the predicted value of  $R$  exceeds the actual value, the Kalman gain will be too low and valuable information may be lost from the observation. Conversely, a smaller value increases the Kalman gain and causes extraction of the unwanted noise as information.

An adaptive approach is taken where the filter observes and attempts to adapt to the actual noise values. While an adaptive Kalman filter has more computation, it does have the ability to both compensate for poor estimates of noise and track non-stationary noise processes.

### C. KALMAN FILTER EQUATIONS

Equation (2.7) illustrates the algorithm of the Kalman Filter. Using a linear recursive formula, the current estimate,  $x_{k+1|k+1}$ , is a linear combination of the previous estimate and the current observation. Because the filter does

not store the observations, it requires a fixed amount of memory to process an arbitrary number of observations. Equations (2.7) and (2.8) are the updated time and updated observation equations, respectively.

$$\hat{x}_{k+1/k} = \Phi \hat{x}_{k/k} \quad (2.7)$$

$$\hat{x}_{k+1/k+1} = (I - G_{k+1}H) \hat{x}_{k+1/k} + G_{k+1}z_{k+1} \quad (2.8)$$

### 1. Kalman Gain

Equations (2.7) and (2.8) show the role of the Kalman Gain,  $G$ , in the state equations. An error function (as described in Sec 2. Error Covariance) is minimized to adjust the values of the Kalman Gain. The Kalman gain is given by Eq. (2.9).

$$G_{k+1} = P_{k+1/k} H^T (H P_{k+1/k} H^T + R)^{-1} \quad (2.9)$$

Subscript  $k$ , used in the equations, shows that  $G$  is a function of discrete time.

### 2. Error Covariance

The error covariance matrices Equations (2.10) and (2.11) are indicators for the magnitude of the estimation error. The matrices are formed from the error of the state vectors.

$$P_{k+1/k} = E[\tilde{x}_{k+1/k} \tilde{x}_{k+1/k}^T] \quad (2.10)$$

$$P_{k+1/k+1} = E[\tilde{x}_{k+1/k+1} \tilde{x}_{k+1/k+1}^T] \quad (2.11)$$

The magnitude of the Kalman gain is determined by the estimate error covariance. The Kalman Filter may be adapted for situations where the expected error changes. A typical example illustrating this feature is a target course change during tracking. Upon detecting the maneuver, there is an increase in the expected error and the Kalman gain, thereby placing more emphasis on the recent observations.

Resetting the error covariance matrix to its initial value,  $P_{0/-1}$ , causes the filter to lock-on the next target observation. Past information on tracking will be disregarded.

### 3. Residual

The residual vector is formed by subtracting the mean from the observed value, Eq. (2.12).

$$r_{k+1} = z_{k+1} - E[z_{k+1}] \quad (2.12)$$

The unbiased estimate consists of zero mean error. This allows the expression of the observation vector as

$$E[z_{k+1}] = E[HX_{k+1}] + E[v_{k+1}] \quad (2.13a)$$

$$= H\hat{X}_{k+1/k} \quad (2.13b)$$

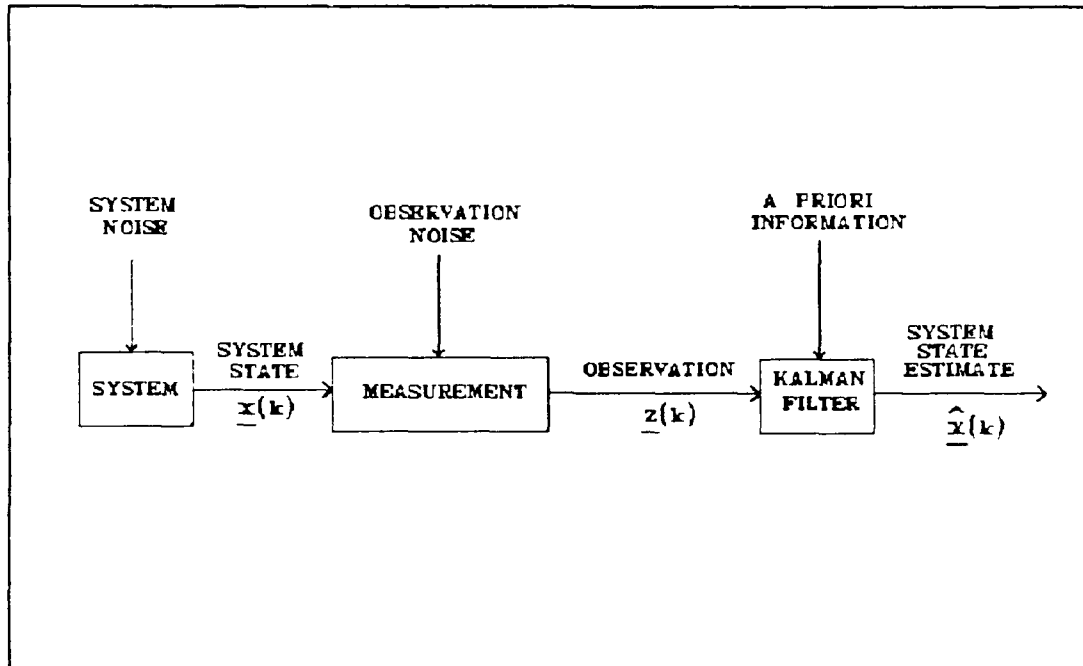
The standart Kalman filter equations are shown in Table 2.2.

**Table 2.2 KALMAN FILTER EQUATIONS**

State equation	$x_{k+1} = \phi_k x_k + \Gamma_k w_k + \Gamma_k u_k$
Measurement equation	$z_k = H_k x_k + v_k$
States estimate	$\hat{x}_{k+1/k} = \phi \hat{x}_{k/k} + \Gamma_k u_k$
Residual	$r_{k+1} = z_{k+1} - H \hat{x}_{k+1/k}$
Error propagation	$P_{k+1/k} = \phi P_{k/k} \phi^T + Q$
Kalman gain	$G_{k+1} = P_{k+1/k} H^T (H P_{k+1/k} H^T + R)^{-1}$
Updated error	$P_{k+1/k+1} = (I - G_{k+1} H) P_{k+1/k}$
Updated states estimate	$\hat{x}_{k+1/k+1} = \hat{x}_{k+1/k} + G_{k+1} r_{k+1}$



Figure 1 summarizes the previously developed math model in a simple block diagram. Important quantities shown are used in estimating the state of the linear system which consists of the system, measurement, and the Kalman filter. Noise factors are included as system and measurement errors.



**Figure 1** Block diagram of system, measurement and estimator

#### 4. Extended Kalman Filter

Whenever system characteristics do not conform to the Linear Time Invariant (LTI) model, the extended Kalman Filter is used. Real systems of concern are nonlinear observation matrices and linear transition matrices.

Equations (2.14) and (2.15) illustrate the state space representation using the nonlinear observation matrix, H. The observation H matrix is a function of the state.

$$X_{k+1} = \Phi X_k + W_k \quad (2.14)$$

$$Z_{k+1} = H(X_{k+1}) + V_{k+1} \quad (2.15)$$

In calculating the observation matrix, H, only the first order terms of the power series expansion of the value of H was maintained.

### III. PROBLEM STATEMENT

#### A. PHYSICAL SYSTEM

In this chapter a mathematical model of the tracking system and physical relationship between the storm and the observer, is introduced. This model is then represented in the state space for use with the Kalman filter equations.

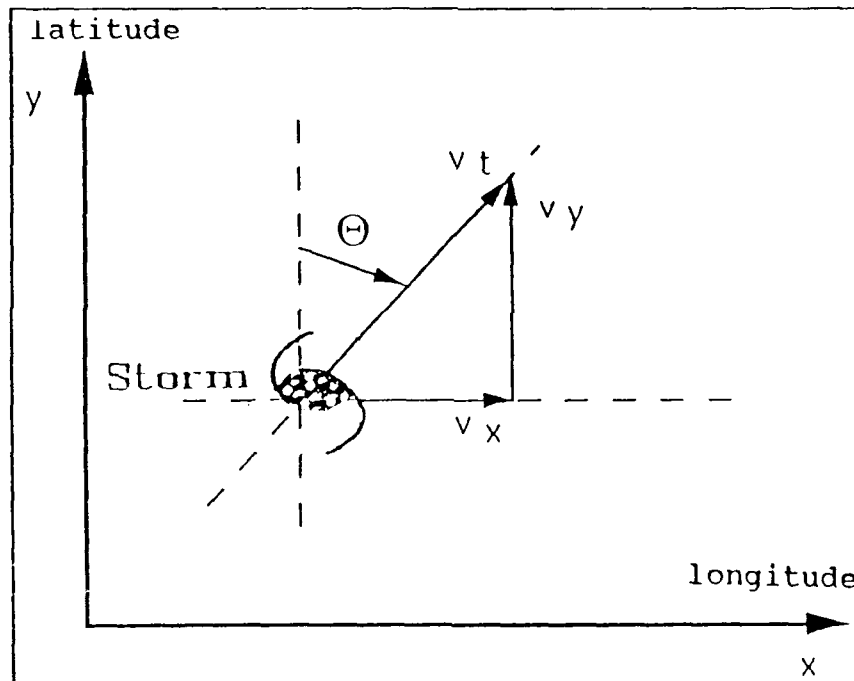


Figure 2 Physical layout of storm

The system uses processes data while tracking a storm. The coordinate system is a two-dimensional cartesian coordinate system. The x and y axis correspond to East and

true North, respectively. The storm is free to move unrestricted throughout the coordinate system.

The position of the storms are given in x (longitude) and y (latitude) coordinates, which are received by a radar or satellite. Estimates are obtained for the location, course, and speed of the storm (the physical states of the plant).

## B. STATE SPACE MODEL

The system to be modelled in this problem is a storm. The discrete-time, state space model of our system is

$$X_{k+1} = \phi X_k + \Gamma U_k + \Gamma w_k \quad (3.1)$$

where

$x_k$  = parameter to be estimated (state vector)

$\phi$  = state transition matrix

$\Gamma$  = system noise coefficient matrix

$U_k$  = deterministic forcing function

$w_k$  = random forcing function.

The state vector  $x_k$  consists of the position and velocity of the storm in Eq. (3.2).

$$x_k = \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix} \quad (3.2)$$

To obtain the estimate, the filter must be initialized with an initial state estimate and an initial error covariance matrix.

An initial velocity is taken to be zero since there is no velocity information at the beginning. The a priori state estimates carry with them a large amount of error. The estimate of this error is used to construct the initial error covariance matrix. The error was assumed to be zero mean and uncorrelated. For these conditions, the initial error covariance matrix is given by

$$P_{0/-1} = \begin{bmatrix} 10^6 & 0 & 0 & 0 \\ 0 & 10^6 & 0 & 0 \\ 0 & 0 & 10^6 & 0 \\ 0 & 0 & 0 & 10^6 \end{bmatrix} \quad (3.3)$$

The matrix  $\phi$  in Eq. (3.1) is chosen to fit the storm's mean dynamics, which moves linearly at constant velocity. The appropriate  $\phi$  matrix is

$$\phi = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.4)$$

The constant,  $T$ , is the observation interval.  $T=6$  hours for purposes of this thesis.

The deterministic forcing functions of the storm are accounted for by the control input vector  $U_k$ . The analysis of

storm motion shows specific external steering flow effecting the storm motion. This effect can be represented in the Kalman filter equations by  $U_k$ .

### **1. Prediction Cyclone Paths**

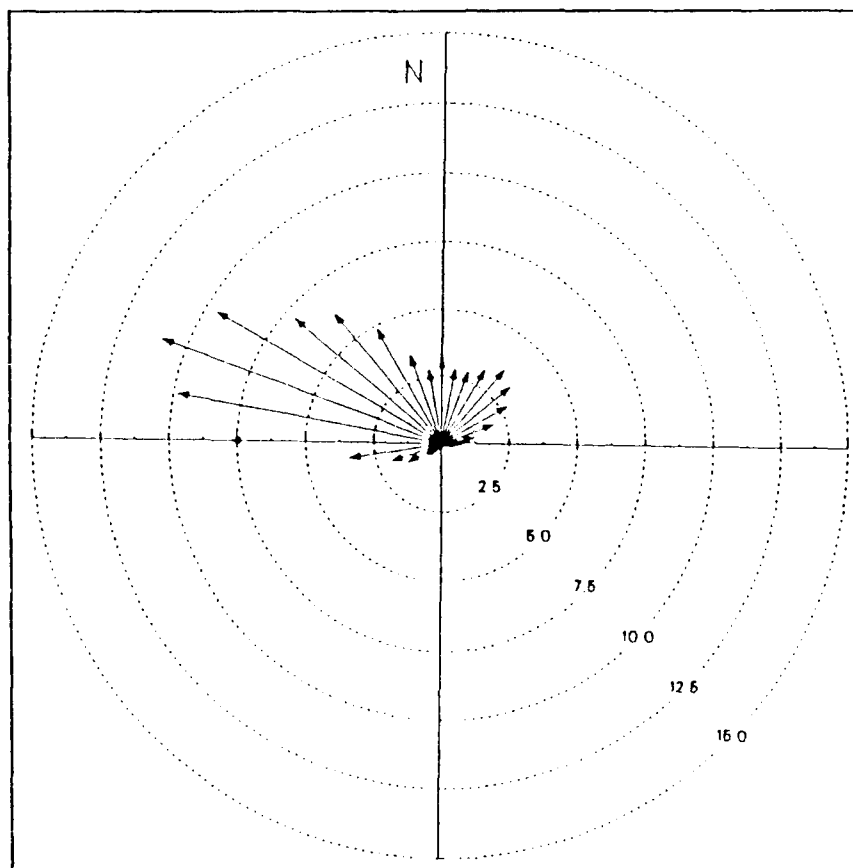
For many years, predictions have been attempted for the paths of tropical cyclones. Steering flow is a method of prediction that has remained popular. Steering flow methods monitor the pressure gradients of the windfield to project the cyclone path vector. This environmental information is used to estimate the two-dimensional (north-south, east-west) advance of the cyclone.

Windfields tend to have small pressure variations in the tropics. Common practice places greater emphasis on the monitoring of two dimensional windfields for making the steering flow path predictions. We will examine these windfields and attempt to identify the characteristic features of different groups of cyclones.

Using tropical cyclones in the Pacific, six-hour position updates are obtained from the Joint Typhoon Warning Center. The path predictions mode will be based on the warning center tracking information.

Historical observations show that Northern Hemisphere cyclones tend to move to the left of the steering currents predicted path. Analytic models by Chan and Holland [Ref.1 pg.107] and others have explained this abnormality due to the

effect of the earth's vorticity. This effect delivers the desired cyclone path, tending to the left of the steering currents. This study will group the cyclones in this area of concern by their direction of movement. Eight years of path data is presented in Figure 3.



**Figure 3** Percentage direction distribution of tropical cyclones

Most cyclones tend toward the 285-295° bearing direction. The distribution forms three groups of cyclones moving westward (265-285°), northward (345-015°), and to the northeast (025-

055°). The most popular directions for movement are clearly the 290-040° directions allowing us to refer to this distribution as bimodal in [Ref.1 pg.107].

To identify the relationship between cyclone path and surrounding pressure forces and windfields, a simple path prediction will be examined. Since this investigation concentrates on the tropics, a standard mercator projection will be used. In the tropic regions grid distortion is slight and insignificant.

Forcing function intensities are mated with one of the directions of motion in Table 3.1. The functions used are  $U_x = \alpha(x - x_0)$  and  $U_y = \alpha(y - y_0)$ . The  $\alpha$  parameter represents the amplitude of the sheer in the two, one dimensional forcing functions. The  $x$  and  $y$  are the longitude and latitude, respectively.

**TABLE 3.1 CLASSIFICATION OF TROPICAL CYCLONE**

Stratification	Intensity( $\alpha$ )
Westward (265-285°)	37
Northward (345-015°)	43
Northeastward (025-055°)	38

Proper choice of coordinate system and neglecting insignificant terms for the simplest description of cyclone



motion, leads to using a cylindrical coordinate system centered at the cyclone center. Additionally, the observance of well established physical laws simplifies the math model cyclone motion to three equations [Ref.6 pg.14].

$$\frac{\partial u}{\partial t} - f v - \frac{v^2}{r} = -\frac{1}{\rho} \frac{\partial p}{\partial r} + F_r \quad (3.5)$$

$$\frac{\partial v}{\partial t} + f u + \frac{u v}{r} = -\frac{1}{\rho r} \frac{\partial p}{\partial \lambda} + F_\lambda \quad (3.6)$$

$$\frac{\partial w}{\partial t} = -\frac{1}{\rho} \frac{\partial p}{\partial z} - g + F_z \quad (3.7)$$

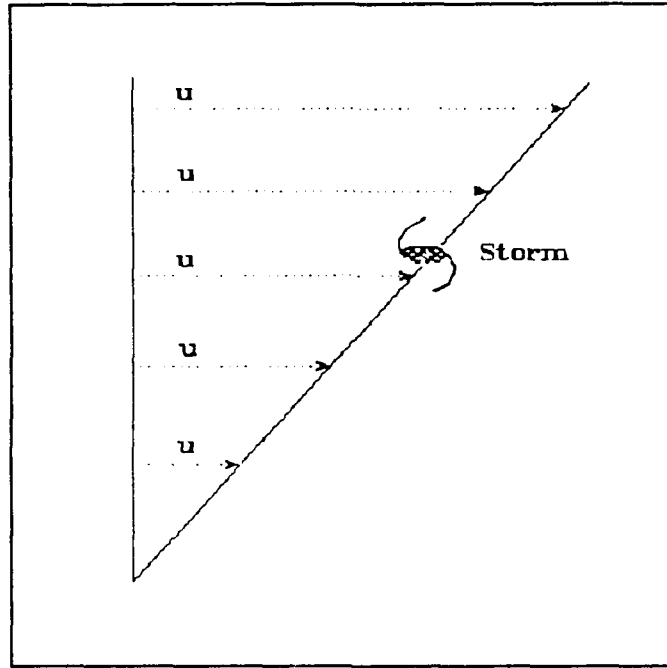
Symbols used represent physical quantities as shown:

u	radial wind component
v	azimuthal wind component
w	vertical wind component
$\rho$	air density
p	air pressure
g	gravity acceleration.

Sources of acceleration in Eq. (3.5) include the Coriolis effect from earth rotation, centripital acceleration  $(1/\rho)(\partial p/\partial r)$  and from local turbulent effects ( $F_r$ ). The nonvarying in time radial wind components (radial wind constant in time  $\partial u/\partial t=0$ ) and friction components allow simplification (frictional effect are negligible  $F_r=0$ ) of Eq. (3.5) to the cylindrical form of the gradient wind equation. This expression is listed by Eq. (3.8).

$$\frac{v^2}{r} + fv - \frac{1}{\rho} \frac{\partial p}{\partial r} = 0 \quad (3.8)$$

It should be emphasized that Eq. (3.8) is the "gradient wind balance" applicable only when the conditions of the above paragraph are met. Pressure gradient force due to differences of pressure within the fluid mass is  $(1/\rho)(\partial p/\partial r)$ . The coriolis force,  $fv$ , acts as a deflecting force normal to the velocity, to the right of the motion in the northern hemisphere and to the left in the southern hemisphere. The centrifugal force in a rotating system, deflecting masses radially outward from the axis of rotation is  $v^2/r$ . The previously mentioned  $U=\alpha(y-y_0)$  or  $U=\alpha(x-x_0)$  represents the large scale environmental steering flow, where,  $U$  is the control input (deterministic forcing function), and  $\alpha$  is the degree of shear in this horizontal forcing function. Eq. (3.8) describes wind distribution within the tropical storm. In summary,  $U=\alpha(y-y_0)$  or  $U=\alpha(x-x_0)$  represents the net large scale environmental high pressure forces acting on the tropical cyclone. The significance of the proposed forcing function,  $U$ , is that it steers the cyclone, whose internal structure is described by Eq. (3.8). ( Note: The letter,  $U$ , is used to denote the forcing function to avoid confusion with the radial wind component,  $u$ .) A schematic description between the forcing function and storm is shown in Figure 4.



**Figure 4** Schematic describing  
between forcing function and storm

## **2. State Excitation Covariance Matrix $Q$**

The unknown accelerations of the storm are accounted for by the state excitation vector  $w_k$ . The analysis of the state excitation covariance matrix  $Q$  accounts for the unknown accelerations of the storm. The results of the derivation are that  $Q$  is given as

$$Q = \Gamma \begin{bmatrix} E(w_x^2) & E(w_{xy}) \\ E(w_{xy}) & E(w_y^2) \end{bmatrix} \Gamma^T \quad (3.9)$$

$$E(w_x^2) = \left(\frac{V_x}{V_t}\right)^2 \sigma_v^2 + V_y^2 \sigma_\theta^2 \quad (3.10)$$

$$E(w_y^2) = \left(\frac{V_y}{V_t}\right)^2 \sigma_v^2 + V_x^2 \sigma_\theta^2 \quad (3.11)$$

$$E(w_x w_y) = E(w_y w_x) = V_x V_y \left[ \left(\frac{\sigma_v^2}{V_t}\right)^2 - \sigma_\theta^2 \right] \quad (3.12)$$

where  $\Gamma$  is

$$\Gamma = \begin{bmatrix} \frac{T^2}{2} & 0 \\ T & 0 \\ 0 & \frac{T^2}{2} \\ 0 & T \end{bmatrix} \quad (3.13)$$

$\sigma_v$  speed deviation of storm

$\sigma_\theta$  directional deviation of storm.

The statistical directional and speed deviations for different directional stratifications were used in the calculation of the state excitation covariance matrix. Several factors may contribute to the scatter of deviations. The noise in the observations and the analyzed field can certainly lead to errors in computing the average flow. If a cyclone is asymmetric, the azimuthally averaged flow will not give a good estimate of the steering flow, and thus may contribute to scatter. The null hypothesis is that the mean directional and speed deviations are both zero. That is, the cyclone moves parallel to the surrounding flow with the same

speed as the flow. For different direction stratification, both the directional and speed deviations are significantly different from zero. The directional and speed deviations values for each group of storm are shown in Table 3.2 from [Ref.1].

**TABLE 3.2 THE DIRECTIONAL AND SPEED DEVIATION VALUES FOR DIFFERENT DIRECTION STRATIFICATIONS**

Directions	265-285	285-345	345-015	015-025	025-055
$\sigma_{\theta}$	28	41	54	45	37
$\sigma_v$	2.3	2.2	2.1	2.1	2.2

The system state equation can be expanded as

$$\begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix}_{k+1} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix}_k + \begin{bmatrix} \frac{T^2}{2} & 0 \\ T & 0 \\ 0 & \frac{T^2}{2} \\ 0 & T \end{bmatrix} \begin{bmatrix} w_x \\ w_y \end{bmatrix}_k + \begin{bmatrix} \frac{T^2}{2} & 0 \\ T & 0 \\ 0 & \frac{T^2}{2} \\ 0 & T \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix}_k \quad (3.15)$$

### C. MEASUREMENT MODEL

For a linear measurement process, the measurements are linearly related to the state variables and can be modelled using the linear measurement equation.

$$z_k = H x_k + v_k \quad (3.16)$$

where

$z_k$  set of measurements

$H$  observation matrix

$x_k$  state vector

$v_k$  measurement noise.

In order to measure longitude and latitude (x,y), the  $H$  matrix must be chosen as follows;

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (3.17)$$

Direct observation of the latitude and longitude provides the x and y coordinates. The measurement equation is

$$\begin{bmatrix} z_x \\ z_y \end{bmatrix}_{k+1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \end{bmatrix}_k + v_k \quad (3.18)$$

#### 1. The Measurement Noise Covariance Matrix R

The measurement noise  $v_k$  has a variance associated with the source of the measurement. This noise is a function of many variables including the time of day, geographical

location, season and frequency. This is generally a non-white Gaussian noise process.

Using the longitude deviation and latitude deviation of the storm, the R matrix is the observation noise covariance matrix. This R matrix accounts for the non white observation noise  $v_k$ ,

$$R = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix} \quad (3.19)$$

where the values used for the longitude deviation ( $\sigma_x^2$ ) since longitudes of storms are equally likely between  $0^\circ$  and  $180^\circ$ . We can say that longitudes of storms are uniformly distributed between  $0-\pi$  radians with an probability of  $1/\pi$ . Because of weather patterns and the necessary conditions for the storm formation, typhoons will generally be uniformly distributed across ocean areas within a certain distance from land. Uniform distribution variance can be express as  $\sigma_x^2 = (a-b)^2/12$ ,  $a=\pi$  and  $b=0 \Rightarrow \sigma_x^2 = \pi^2/12$ .

Distribution in latitude, however, will be predictable within a natural or Gaussian distribution. The values ( $\sigma_y^2$ ) are used for the latitude deviations of storms. For Gaussian distribution, variance  $\sigma^2 = E^2 - (E)^2$  where  $(E)$  is mean and  $E^2$  is second moment. Each tropical cyclone position is assigned a position code number (PCN) to indicate the accuracy of the fix position. The latitude deviations of storms are given in

Reference 5 for different parts of the world. They depend on PCN. Table 3 shows the latitudes and longitudes deviations of the Northwest Pacific storms. The user must be careful to find correct values of  $\sigma_y^2$  for the part of the world he is concerned with.

**TABLE 3.3 THE MEASUREMENT NOISE COVARIANCE MATRIX VALUES**

PCN	SATELLITE DERIVED		RADAR DERIVED	
1 or 2	$\sigma_x^2 = \pi^2 / 12$	$\sigma_y^2 = 184.9$	$\sigma_x^2 = \pi^2 / 12$	$\sigma_y^2 = 361$
3 or 4	$\sigma_x^2 = \pi^2 / 12$	$\sigma_y^2 = 292.4$	$\sigma_x^2 = \pi^2 / 12$	$\sigma_y^2 = 361$
5 or 6	$\sigma_x^2 = \pi^2 / 12$	$\sigma_y^2 = 948.6$	$\sigma_x^2 = \pi^2 / 12$	$\sigma_y^2 = 361$

#### D. SMOOTHING ALGORITHM

Smoothing attempts to improve the accuracy of past state estimates using information from past and current observations. This offline procedure has many variations. This thesis is concerned with Fixed Interval Smoothing only. As the name implies, fixed interval smoothing requires a finite memory capacity to smooth each state estimate over a fixed time interval. All observations before and after the estimate time are within the interval, used by smoothing algorithm. The smoothing algorithm begins with the most recent filter estimate and works backwards in time. One



repetition of the algorithm is performed for each subscripted state estimate. For our fixed interval smoothing algorithm operating on an interval  $k$  units in duration,  $k-1$  repetitions of the algorithm are performed during each smoothing.

$$A_k = P_{k/k} \Phi^T P_{k+1/k}^{-1} \quad (3.20)$$

$$\hat{x}_{k/N} = \hat{x}_{k/k} + A_k (\hat{x}_{k+1/N} - \hat{x}_{k+1/k}) \quad (3.21)$$

$$P_{k/N} = P_{k/k} + A_k (P_{k+1/N} - P_{k+1/k}) A_k^T \quad (3.22)$$

where

$A_k$ =smoothing filter gain matrix,

$\hat{x}_{k/N}$ =smoothed state estimate a time  $k$  based on  $N$  observation

$P_{k/N}$ =smoothed state error covariance matrix.

#### IV. COMPUTER SIMULATIONS

##### A. GENERAL

The Kalman filter program STORM.FOR has been used by Asim [Ref.2] in development of a Kalman filter for storm tracking. The STORM.FOR program was modified to account for the complex effect of pressure forces in the storm. Graphical results were obtained using the Matlab graphics package and the plots included are representative of the results obtained from the three different storms.

Three different groups of storms (1988) are simulated using a program given in the Appendix. The storm tracks used were obtained from data collected at the Joint Typhoon Warning Center (JTWC), while the position coordinates were obtained using satellite and radar. There were three types of data: raw data (observations), best track data and predictions. The raw data was processed just as if it were the real-time observations of the hurricane to produce the filtered estimations. This is the input file for the filter and smoothing algorithm. STORM.FOR generates FILDATA.DAT and SMDATA.DAT contains the track information.

Three typhoons, Hal, Uleki and Doyle were simulated using the information obtained from the JTWC.

## 1. Typhoon Hal

An alert was superseded by the warning of a tropical depression which was assigned the name Hal [Ref.5]. Later this was upgraded (081200Z) to tropical storm Hal. Initially, Hal tracked west southwestward, but eventually settled into a west northwestward track.

Earlier at 101200Z, when Hal was 120 nm (222 km) northeast of Maug in the northern Marianas, the tropical cyclone started to decelerate and track to the southwest in response to a strong pressure ridge to the north and west. After typhoon Hal reached its peak intensity of 105 kt (54 m/sec) at 111200Z, it continued onward and passed over Maug. Power outages and minor property damage were reported on the island of Guam. With a mid latitude trough creating lower pressure in the subtropical ridge north of the typhoon, Hal's direction of track changed to the north-northwest. At 150000Z, Hal approached 32 degrees north latitude and started to curve and accelerate. Hal's widespread destructive force caused several deaths and injuries along the coastal areas near Tokyo. As Hal moved off to the northeast, its central convection was stripped away from its low level circulation center consequently weakening the system.

Figure 5 shows typhoon Hal's best track. The typhoon's track data is in six-hourly increments. The filter and filter with smoothing tracks are shown in Figure 6 and Figure 7, respectively. Figure 8 shows the track results obtained with

the Kalman filter and after the addition of the smoothing algorithm. The filter average tracking error stands at approximately 1 nm, while the smoother average tracking error is always less than 1 nm. Figure 9 shows the tracking errors of the filter and smoother. It is observed that use of the smoother reduces the sensitive to large course changes.

These significant tracking error reductions were generated by heuristics for this thesis. The reductions are the result of an improved filtering estimation process outlined in Chapter III. The improvements were made in the R matrix (observation noise covariance matrix), the Q matrix (the input noise covariance matrix), and the addition of  $u_k$ , a forcing function, to the state estimate equation.

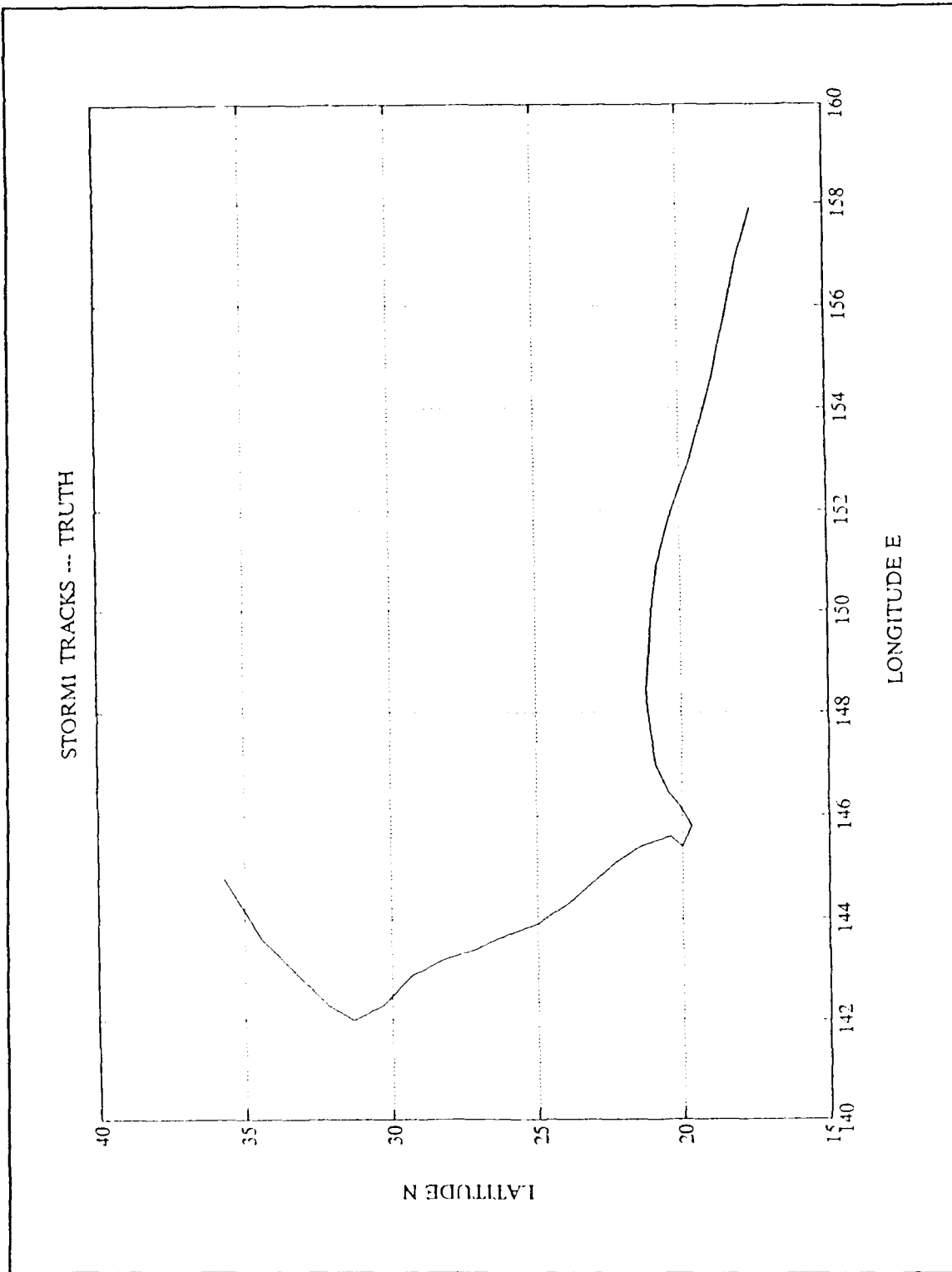
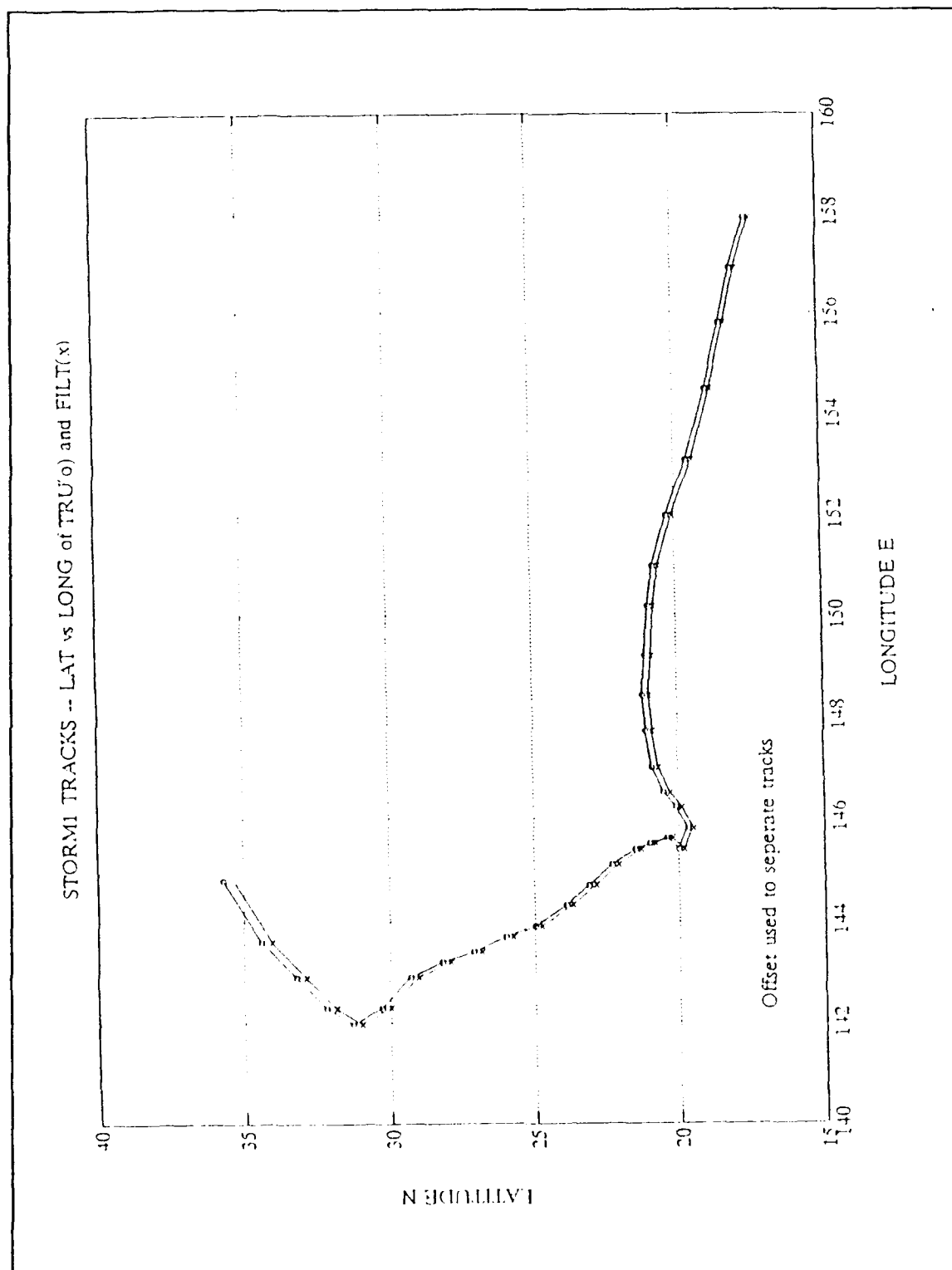


Figure 5 The Best Track of Typhoon Hal



**Figure 6** Filtered Track of Typhoon Hal

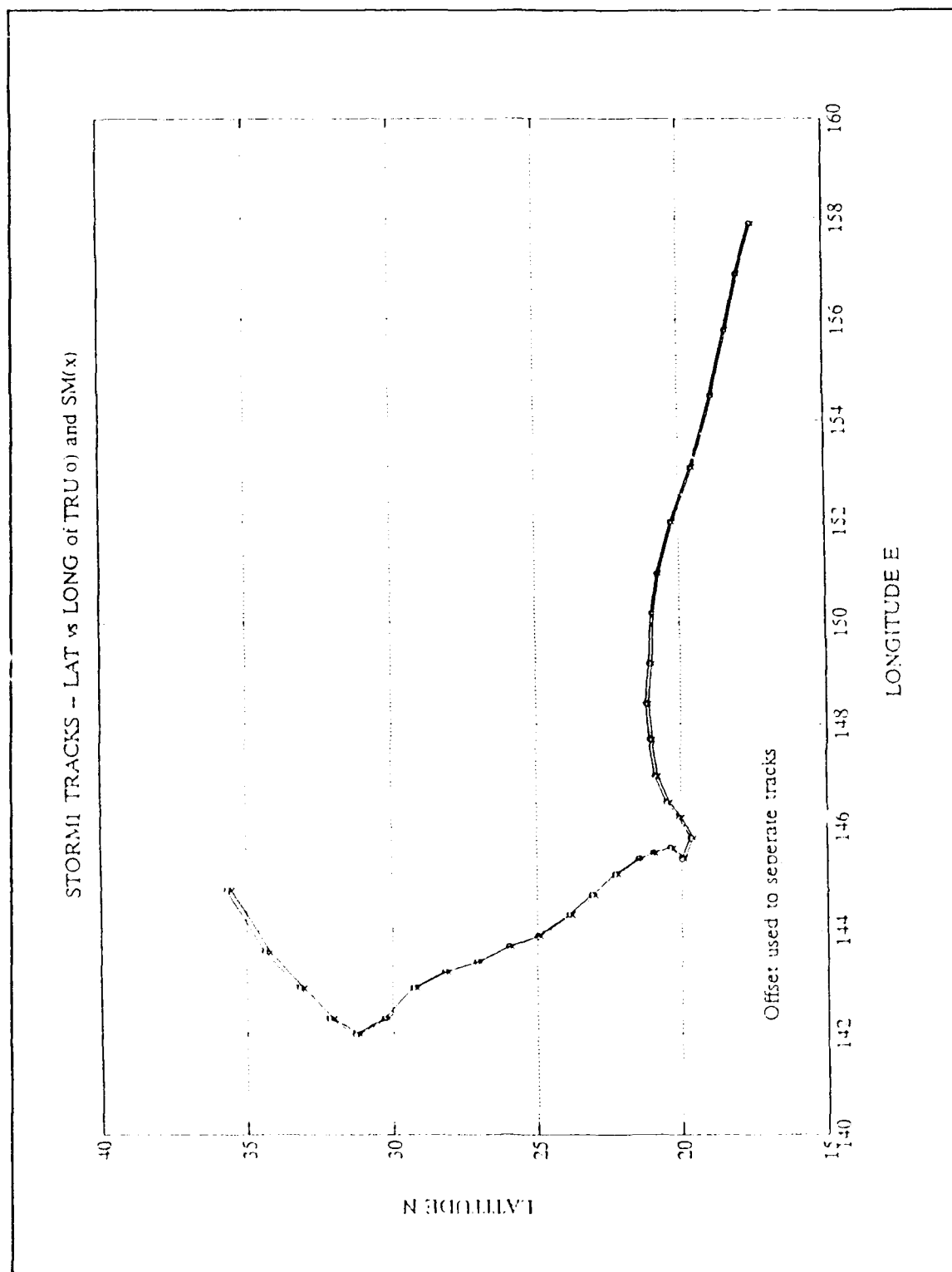


Figure 7 Smoothed Track of Typhoon Hal

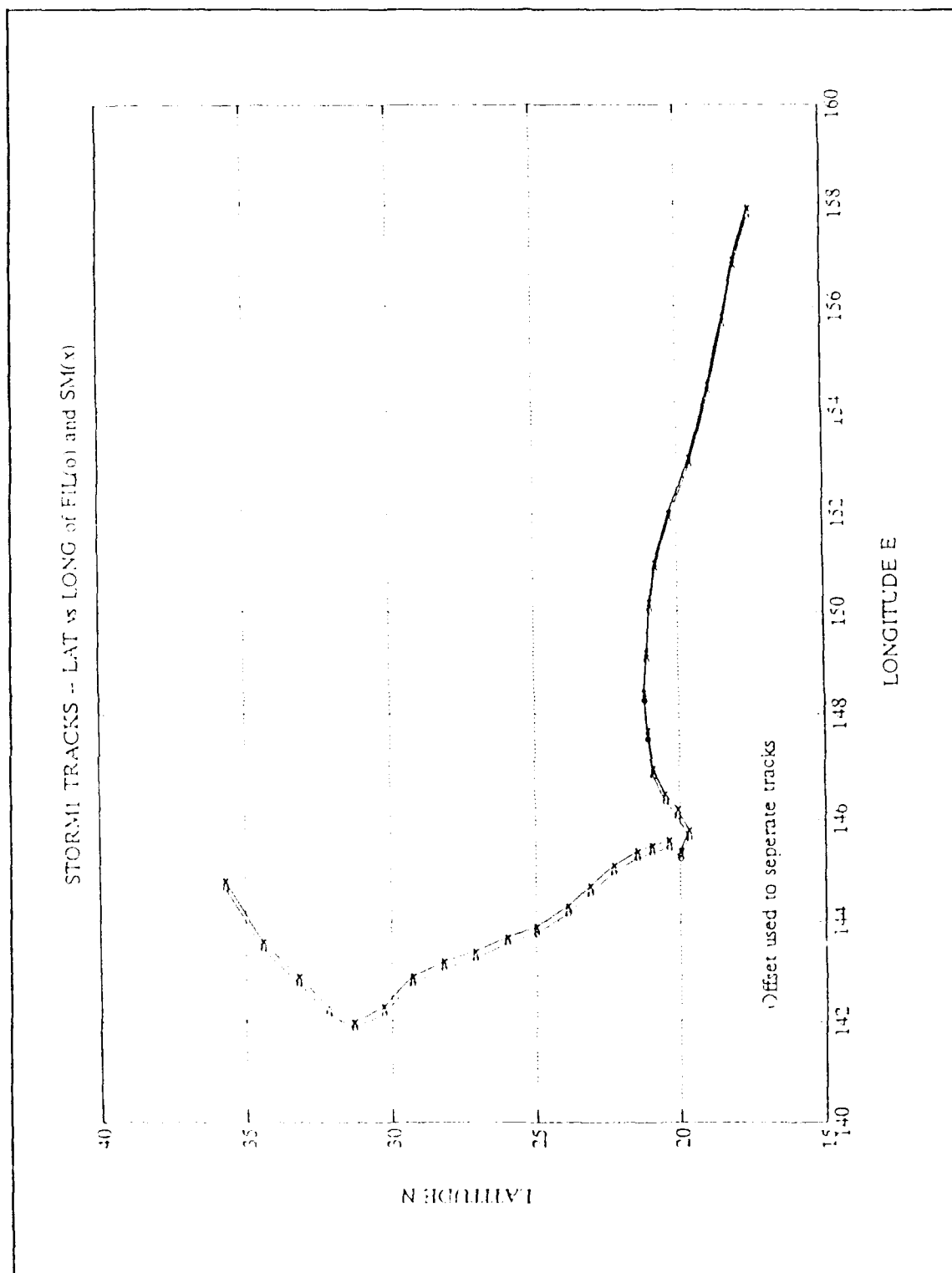
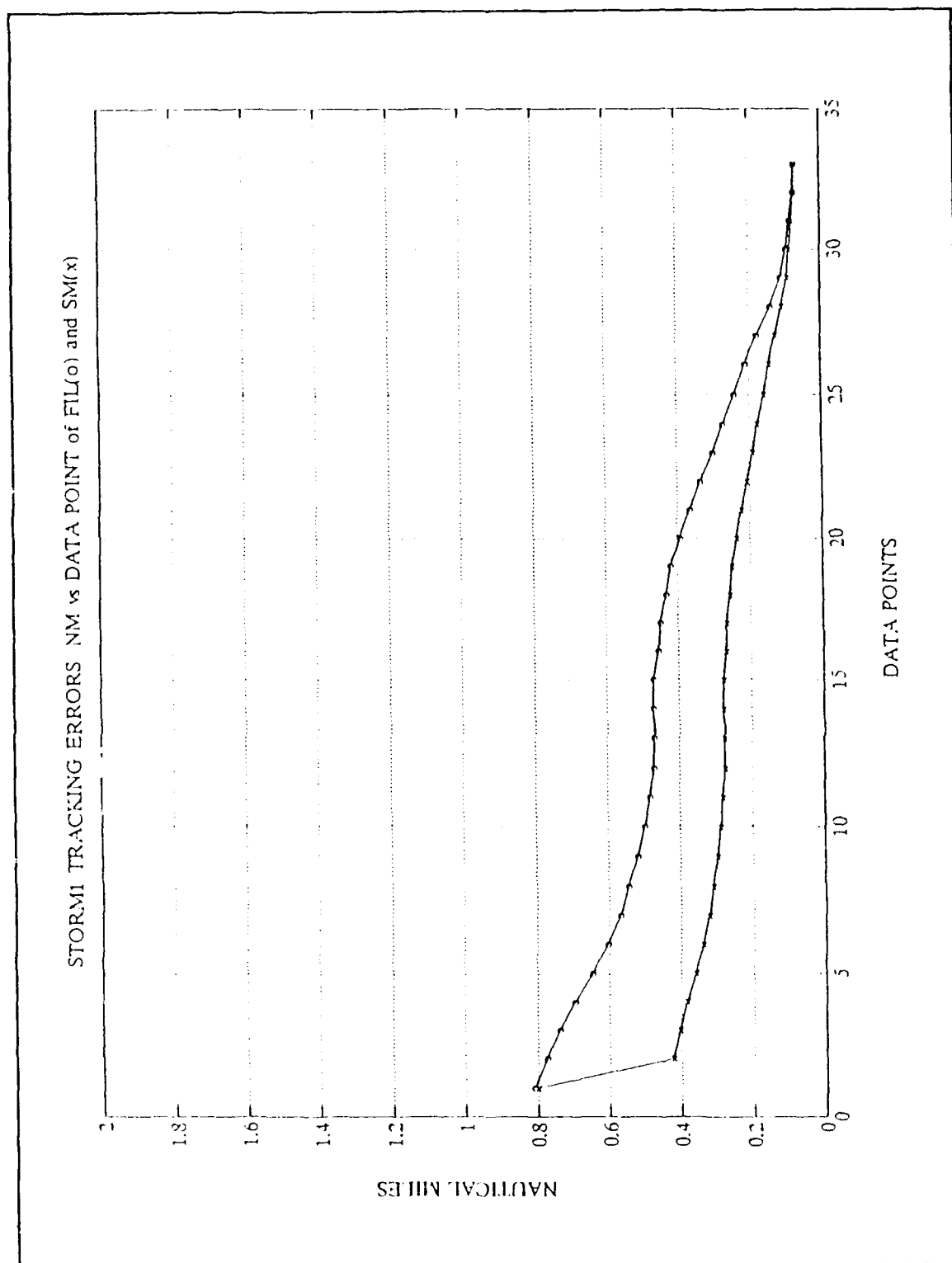


Figure 8 Filtered and Smoothed Track of Typhoon Hal





**Figure 9** Tracking Errors of Filter and Smoother for Typhoon Hal

## 2. Typhoon Doyle

The best track of typhoon Doyle is shown in Figure 10. These best track positions are in six hourly increments. The satellite intensity estimate was 40 kt (21 m/sec) maximum wind speed at 151200Z. At first warning, the system was 96 nm (178 km) east-northeast of Wake Island. For the 24-hour period from 151800Z to 161800Z, intensity increased from 50 to 115 kt (26 to 59 m/sec). Doyle peaked in intensity at 161800Z and assume a northward track at 170000Z. Doyle cut a curvy path following lower pressures between the high pressure subtropical ridge to the southeast and another high cell to the northwest centered near 42 degree north latitude. After gradual weakening, Doyle began to move into a northeast semicircle at 180900Z. Figures 11 and 12 show Doyle's path as it slowed and moved between two high-pressure ridges as Doyle moved northeast Kalman Filter position estimates where formulated. These estimates with smoothing applied, appear in Figure 13.

In general, the smoother increased the accuracy of tracking. Figure 14 was plotted using the tracking error of the filter together with the smoother. The average tracking errors for this storm are 1 nm for the filter and smoother estimate. In comparing the best track and the filter estimate, they virtually duplicate each others tracks. This improved accuracy was achieved by using time varying values for the R and Q matrices and the addition of a forcing

function to the state estimation equation. These time varying values were determined by the storm speed, direction, range deviation and bearing deviation. These results illustrate the improved capability of a Kalman Filter using the time varying calculation of parameters.

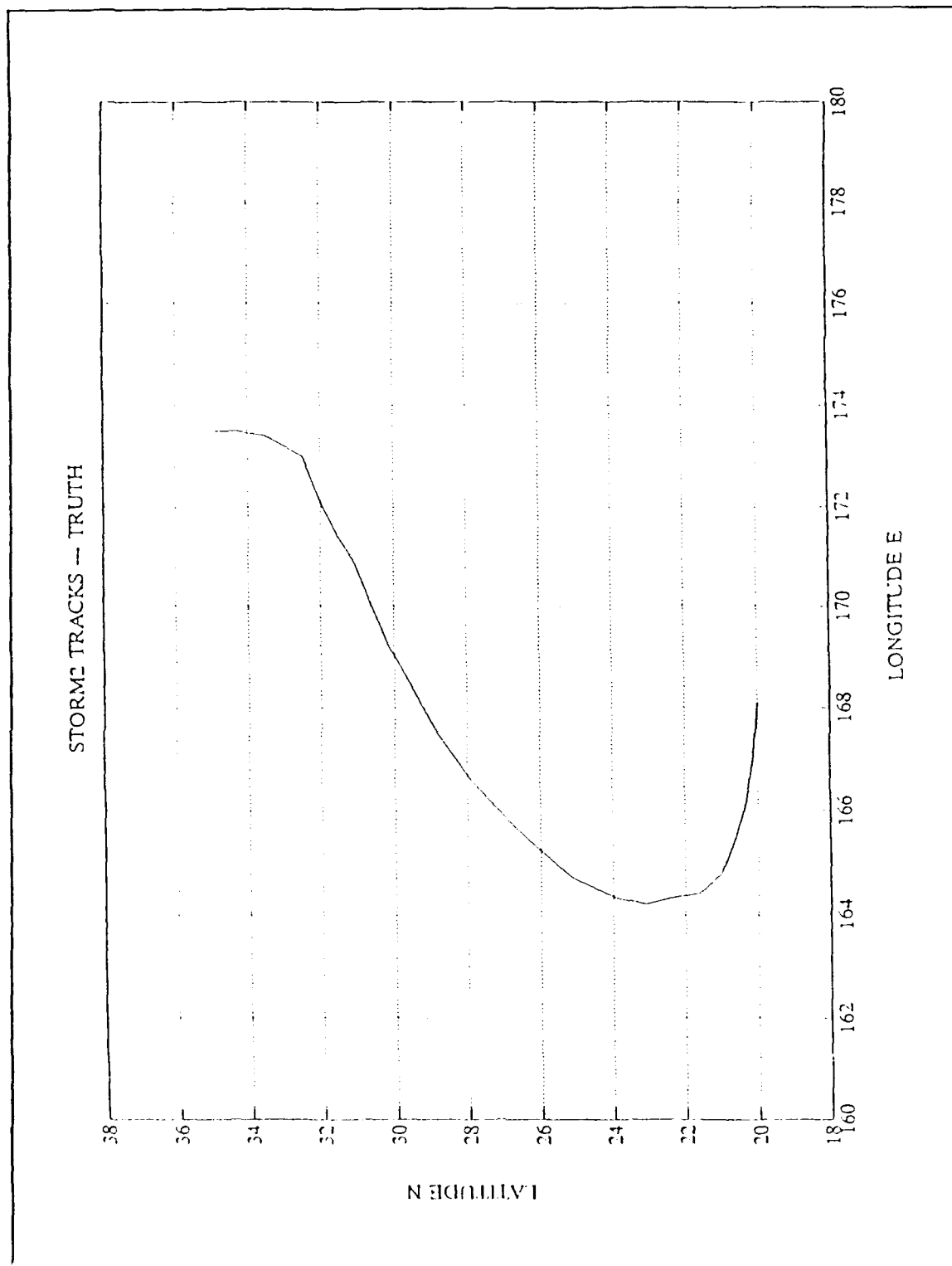


Figure 10 The Best Track of Typhoon Doyle

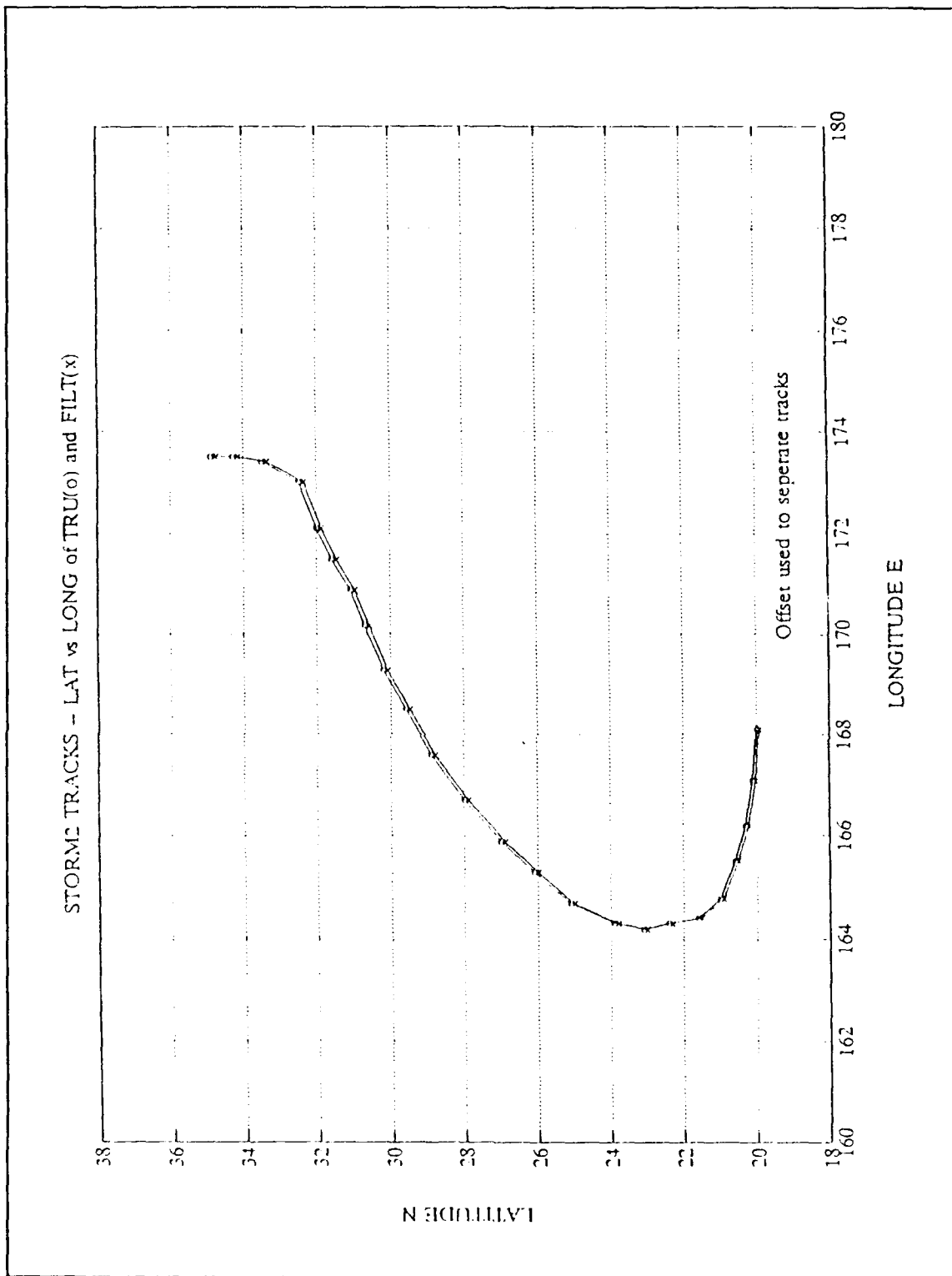


Figure 11 Filtered Track of Typhoon Doyle

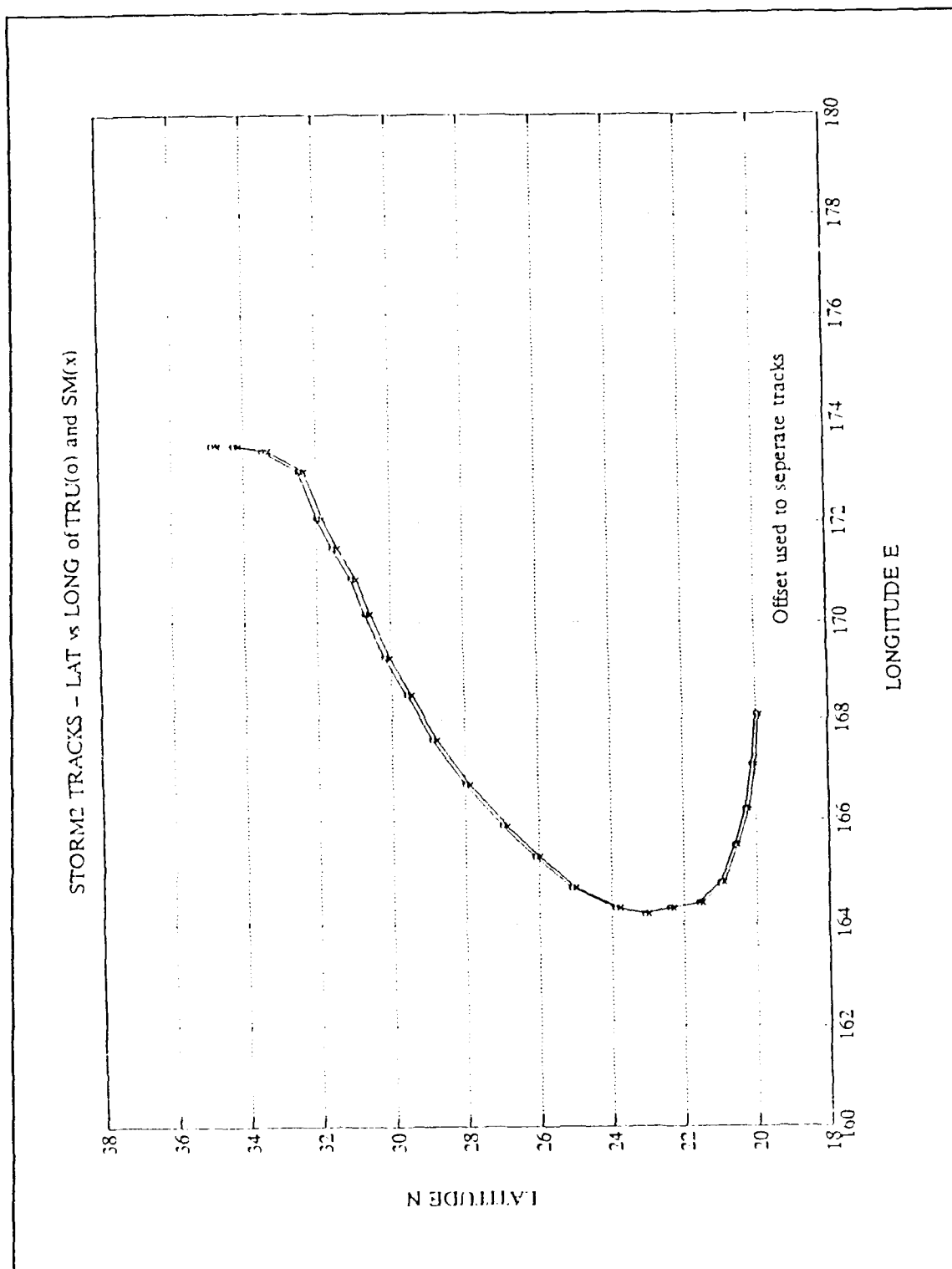


Figure 12 Smoothed Track of Typhoon Doyle

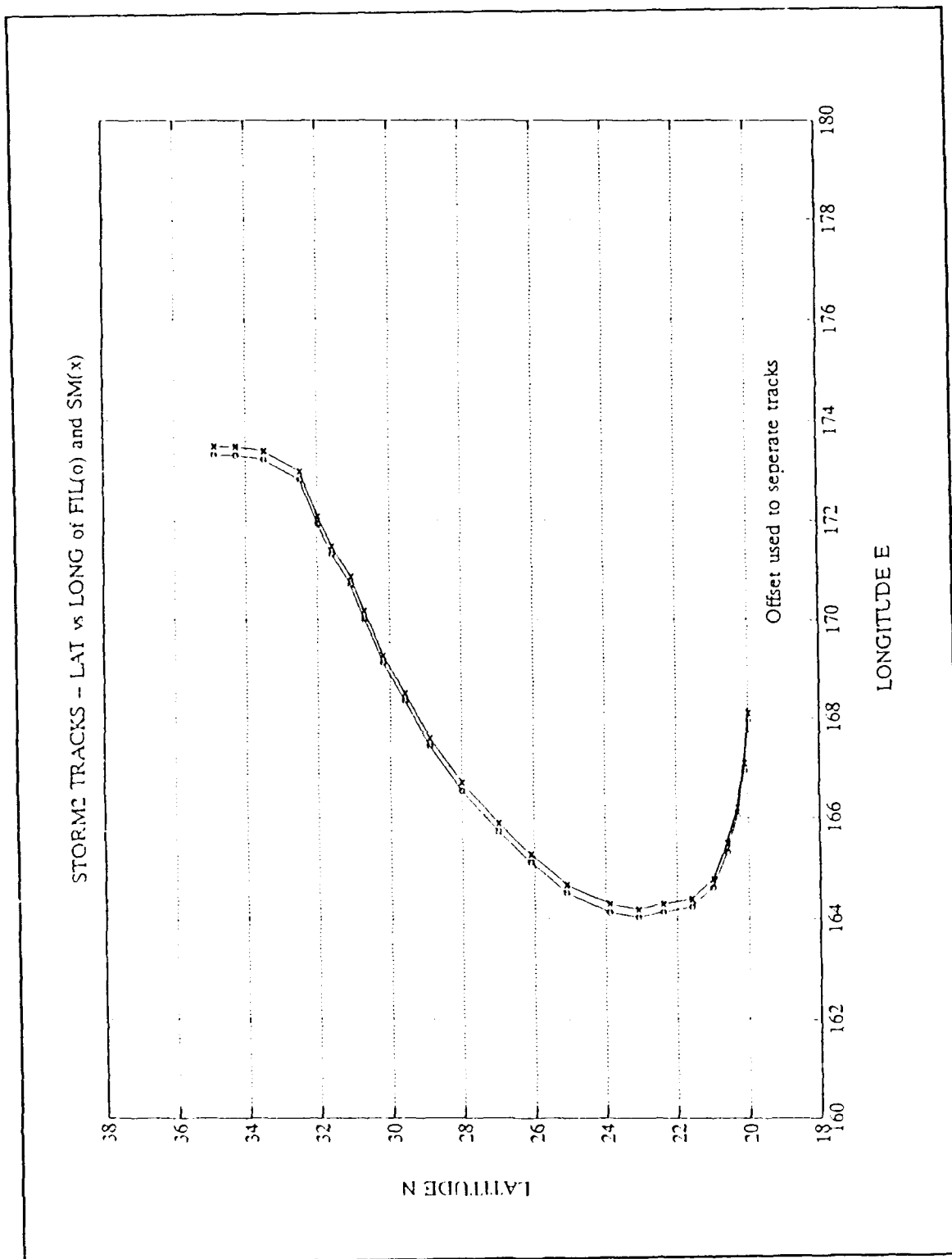
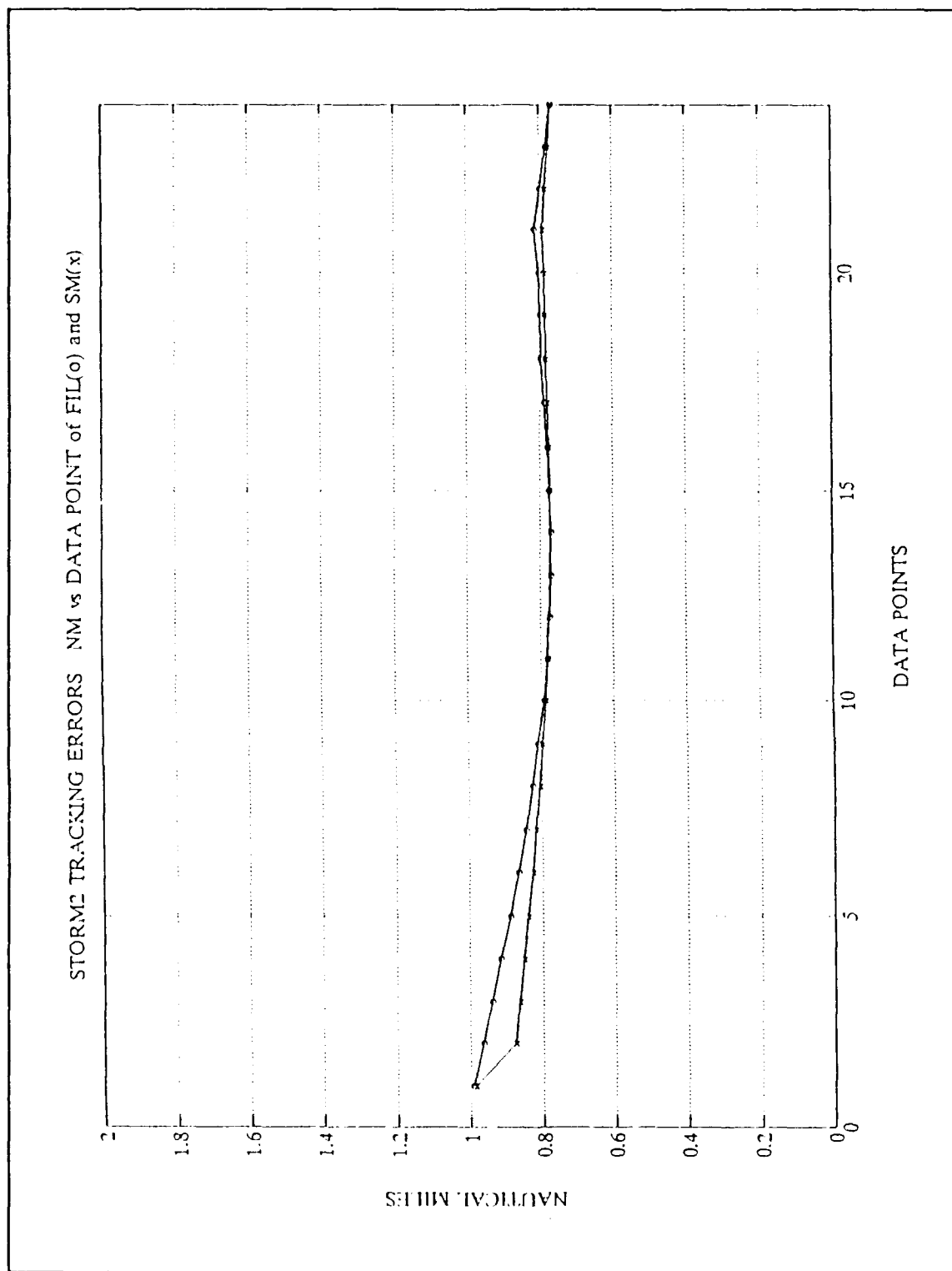


Figure 13 Filtered and Smoothed Track of Typhoon Doyle



**Figure 14** Tracking Errors of the Filter and Smoother for Typhoon Doyle



### 3. Typhoon Uleki

Uleki was first detected at 281800Z August 1988. During the next four days, Uleki tracked westward and intensified. At 291800Z Uleki had reached tropical storm intensity. As Uleki approached the Hawaiian Islands at peak intensity, the direction of movement changed from west-northwestward to northward. The hurricane approached to within 270 nm of Honolulu at 040000Z before changing course to the westnorthwest and accelerating. The tropical cyclone began a weakening trend as it entered a shearing environment. Uleki continued to move west-northwestward and approached the International Dateline. At this time (080600Z), the tropical cyclone had an intensity of 90 kt (46 m/sec). Uleki passed onward to the westnorthwest along the southern edge of a subtropical ridge, and gradually slowed. At 100600Z, the speed of movement had dropped from 15 kt to 6 kt. The typhoon had entered a low pressure steering flow in an area between two high pressure subtropical ridges with a mid-latitude trough approaching from the west; Uleki then began a 'step climb' to the north-northwest. Uleki returned to a smooth northwestward track and weakened.

Again, the best track vs. filtered track and filtered vs. smoothing tracks, are almost identical in Figures 15 through 19. As before the tracking error is very small in magnitude. This employment improved the parameters of the tracking estimate. The Kalman filter has illustrated impressive

results under three different tracking conditions. Tracking graphs have shown that the three analyzed storms traversed different directions along paths of varying complexity.

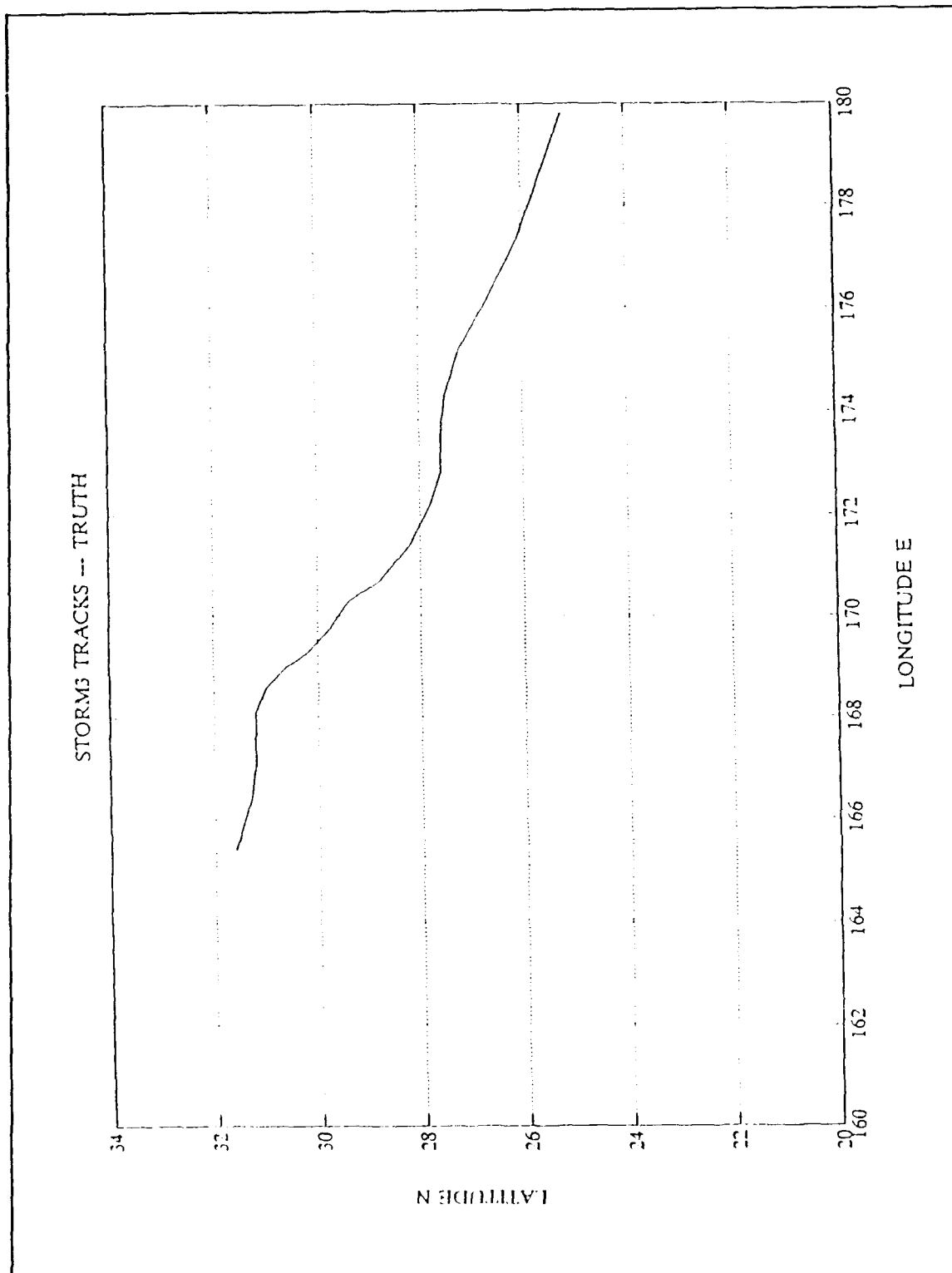


Figure 15 The Best Track of Typhoon Uleki

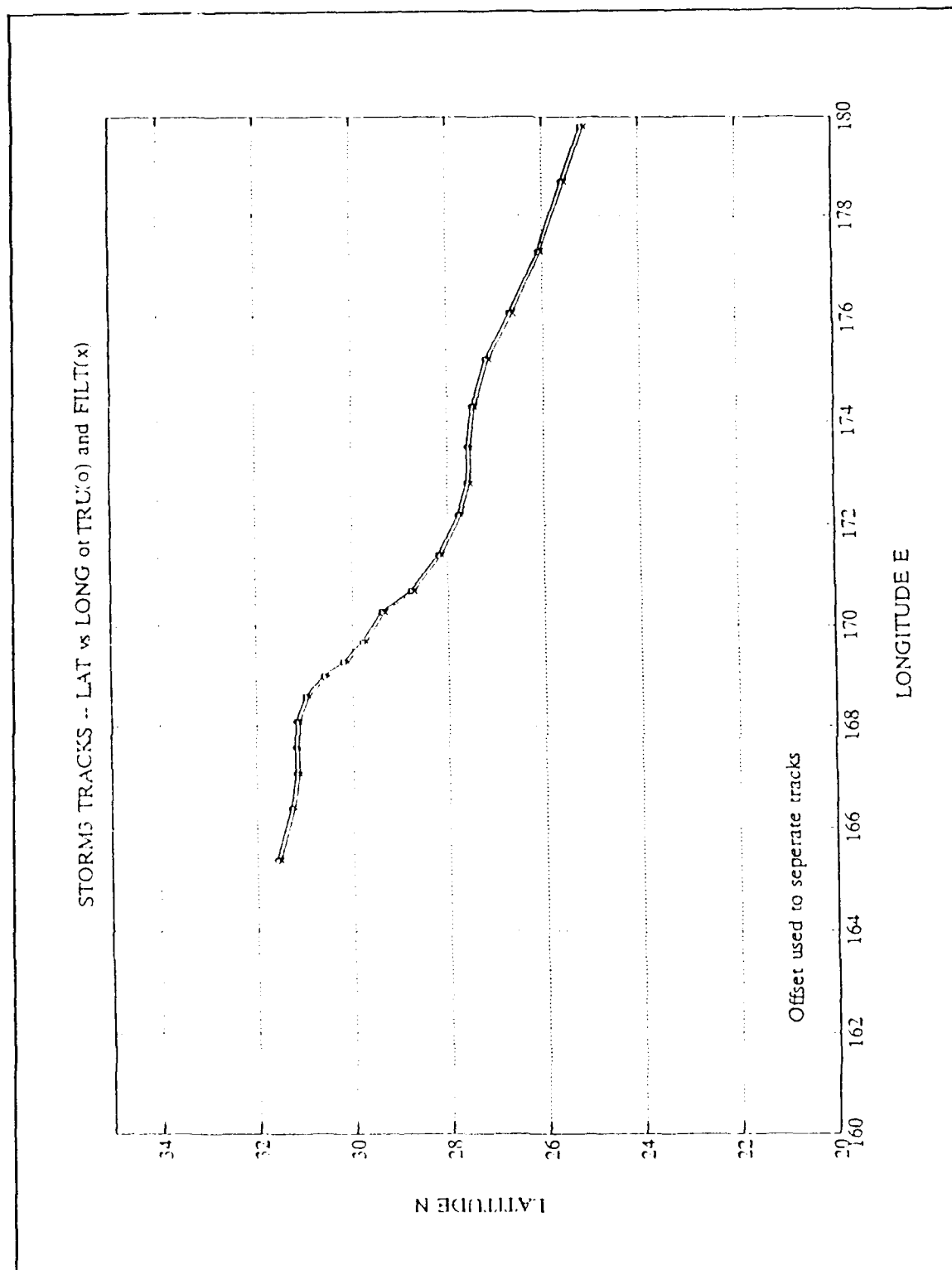


Figure 16 Filtered Track of Typhoon Uleki

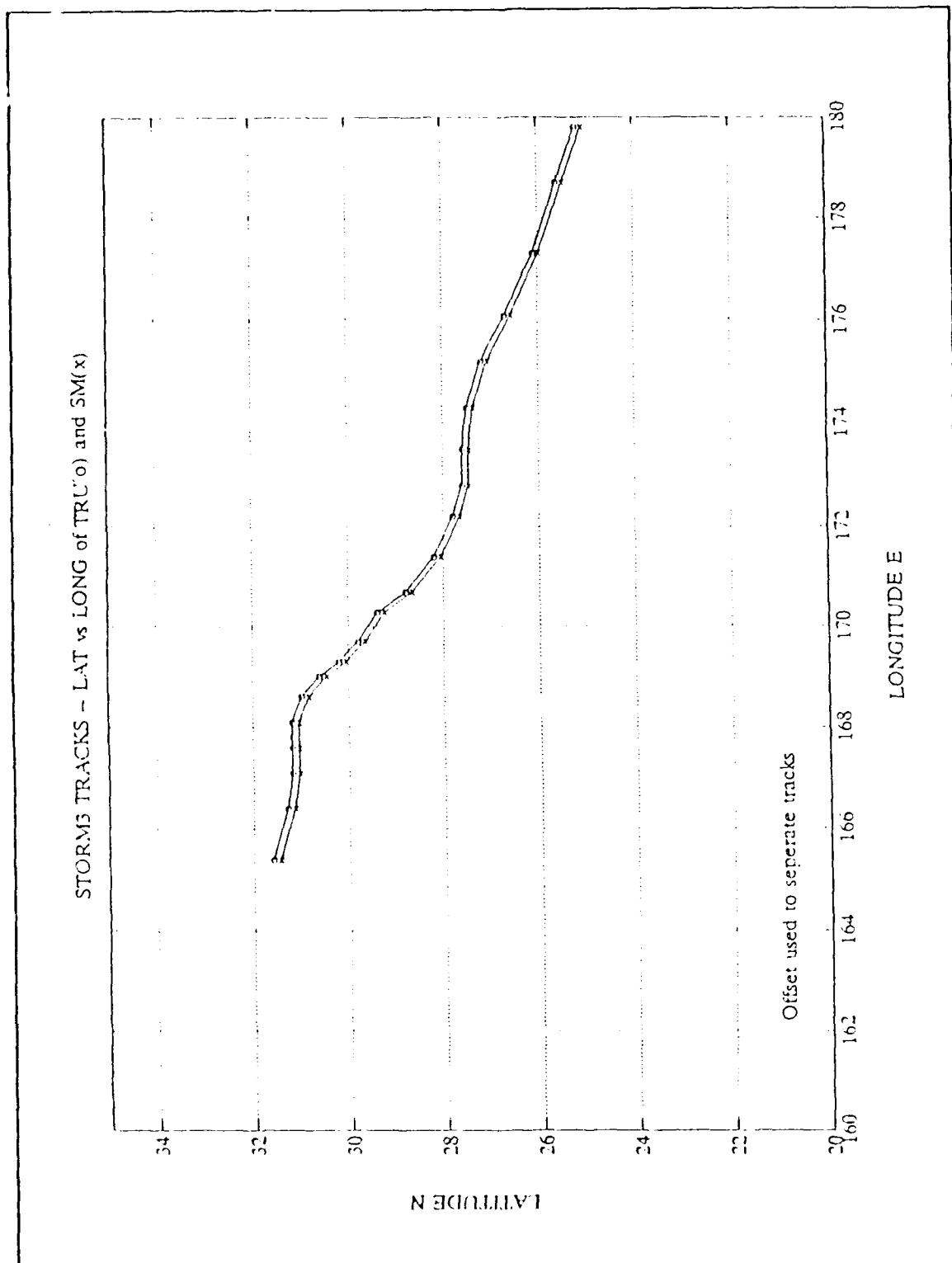


Figure 17 Smoothed Track of Typhoon Uleki

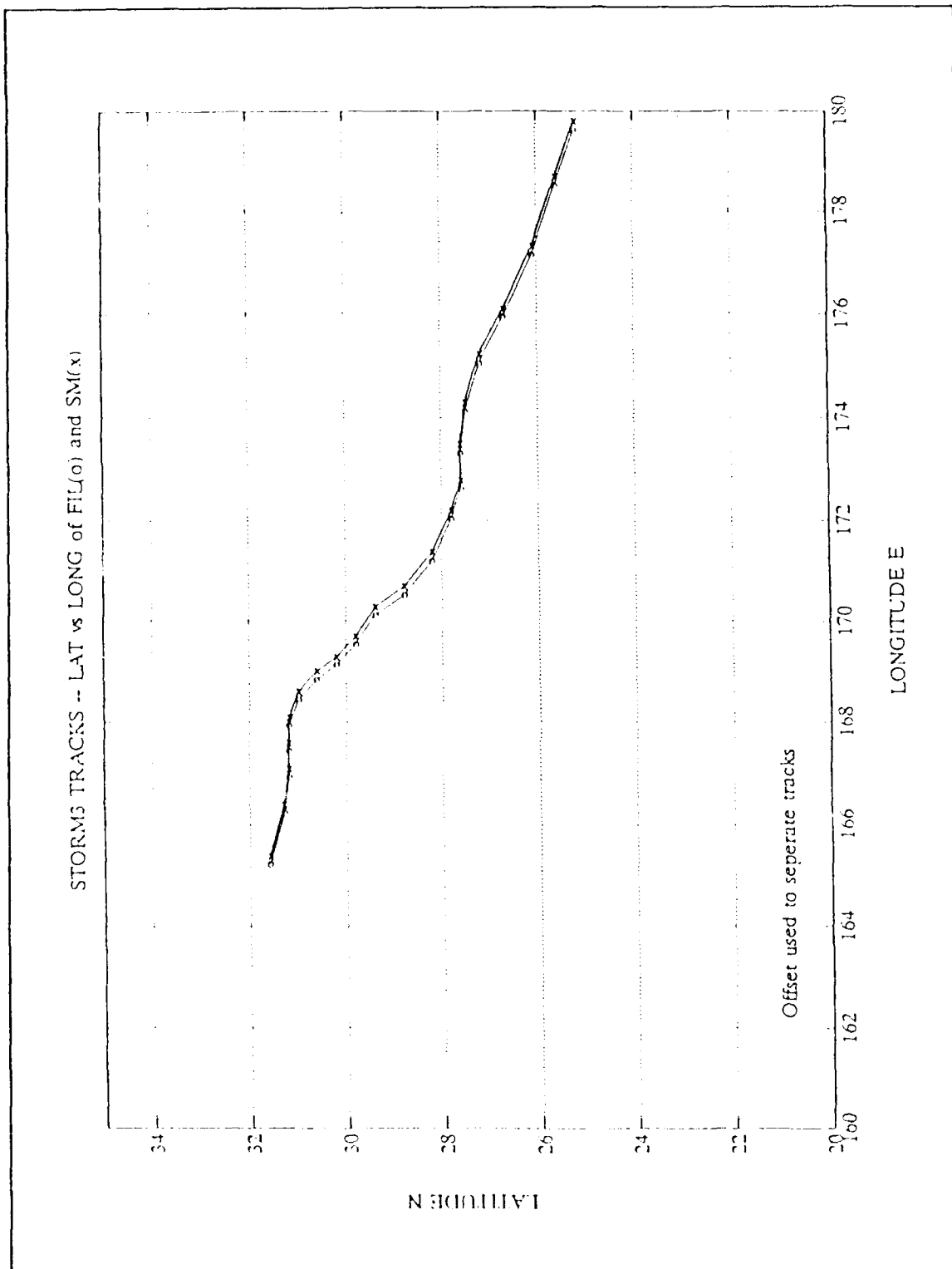


Figure 18 Filtered and Smoothed Track of Typhoon Uleki

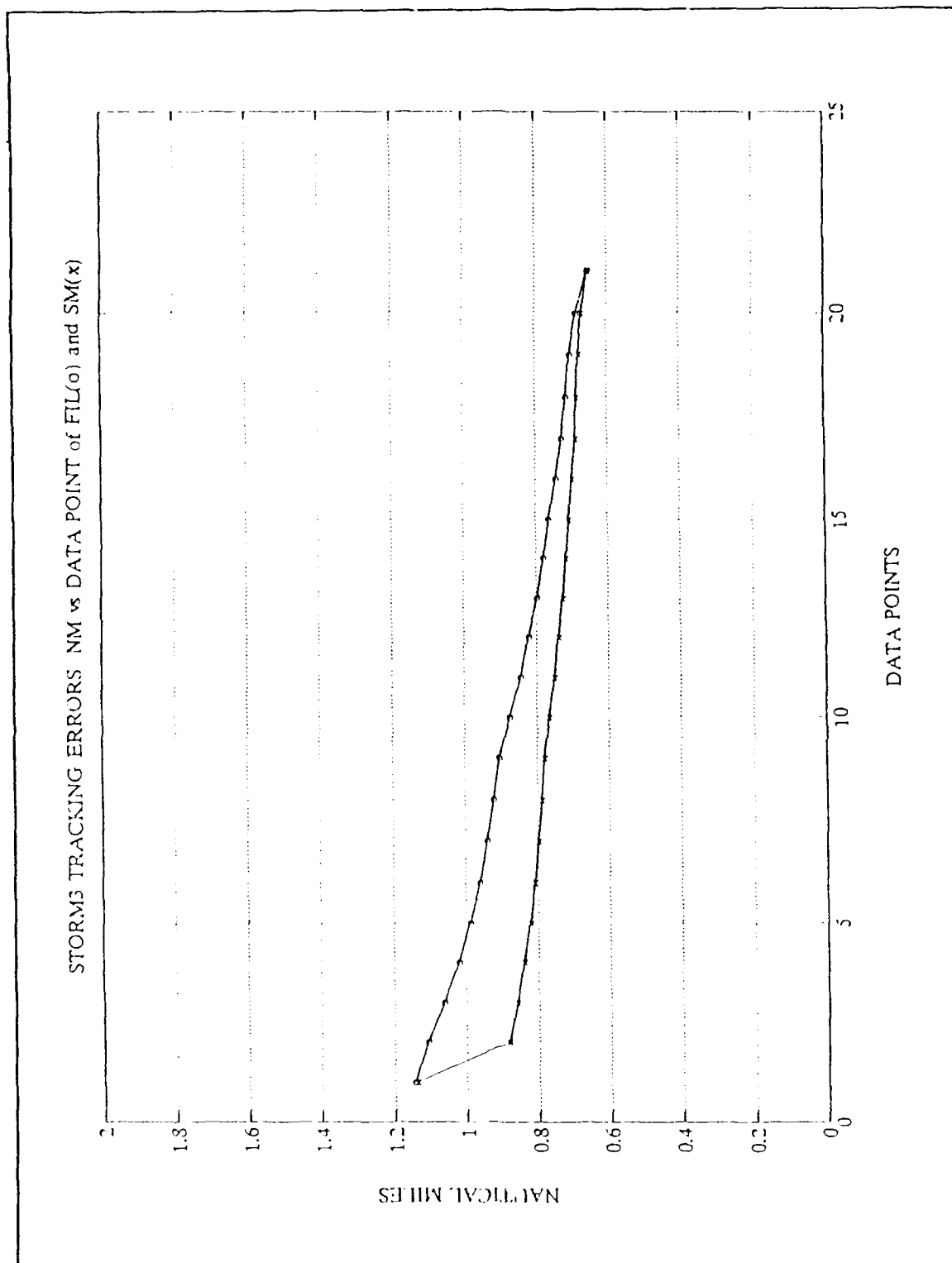


Figure 19 Tracking Errors of the Filter and Smoother for Typhoon Uleki

## V. CONCLUSION

The purpose of this research was to continue previous work in the area of storm tracking using Kalman Filter techniques. A fixed interval smoothing algorithm, developed in past research, was used to improve the overall accuracy of the storm tracking capability. Three different tropical storms were simulated and the accuracy of the observed, filtered and smoothed storm tracks were analyzed and discussed.

The smoother did improve the track accuracy on the basis of the best track storm position.

In previous research, parameters in the  $Q_k$  (state excitation covariance matrix) and  $R_k$  (measurement noise covariance matrix) matrices were established by curve fitting. This thesis achieved more accurate and more stable results adaptively using observation and input variance related noise amplitudes to set the  $Q_k$  and  $R_k$  matrix parameters. Additionally, the state estimate equation for position and velocity was altered to include a forcing function based on steering flow or pressure ridges surrounding the storm.

By estimating the noise power from the variance and adapting the filter to compensate for varying noise power and applying the external force to the system, the performance benefits were significant. However, much work needs to be



done in this area to improve the noise power estimate and external force estimate further, so that the Kalman filter can provide still better state estimates.

## APPENDIX STORM.FOR SOURCE CODE

```

C *** STORM ***

C***** TO RUN *****
C
C 1) ENSURE STORM DATA IS AVAILABLE
C 2) RUN STORM.FOR
C 3) COPY OBSDATA,FILDATA,SMDATA,ERRDATA -->MATLAB SUB-DIR.
C 4) BEGIN MATLAB --> RUN STORM.M
C
C *****
C THIS PROGRAM EMPLOYS AN ADAPTIVE EXTENDED KALMAN FILTER
C WITHE A FIXED INTERVAL SMOOTHING ALGORITHM TO TRACK A
C TROPICAL STORM USING OBSERVED LATITUDES AND LONGITUDES.
C *****

C ***VARIABLE DEFINITIONS***

C AK          = SMOOTHING FILTER GAIN MATRIX
C AKT         = TRANSPOSE OF AK
C BRG         = MEASURED TARGET BEARING IN RADIANS
C BRKKM1      = PREDICTED TARGET BEAR MEASUREMENT IN
C              RADIANS BRG(K|K-1)
C DBRG        = MEASURED TARGET BEARING IN DEGREES
C DT          = TIME DELAY BETWEEN OBSER.T(K) - T(K1)
C DTOR        = DEGREE TO RADIAN CONVERSION FACTOR
C E1,E2       = MEASUREMENT RESIDUAL, Z(K) - H(X(K|K-1))
C E1M1,E2M1   = MEASUREMENT RESIDUAL AT PREV.OBSERVATION
C E1M2,E2M2   = MEASUREMENT RESIDUAL TWO OBSER.PREVIOUS
C FAC1        = RECIPROCAL OF VARE
C G           = KALMAN GAIN VECTOR
C GATE1       = 1.5*STANDARD DEV.OF RESIDUAL PROCESS
C              USED AS A GATE IN MANEUVER DETECTION
C H           = MEASUREMENT MATRIX
C HDG         = ESTIMATED TARGET HEADING IN DEGREES
C HT          = TRANSPOSE OF H
C I           = COUNTER
C IMAT        = 4 X 4 IDENTITY MATRIX
C J           = COUNTER
C K           = ITERATION INTERVAL
C LPKK        = STATE COVARIANCE MATRIX.AFTER PREV. OBS.
C LPKKM1      = A PRIORI STATE COVARIANCE ESTIMATE
C LXKK        = STATE ESTIMATE AFTER PREVIOUS OBS.
C LXKKM1      = A PRIORI STATE ESTIMATE
C M1,M2       = AVERAGE OF RESIDUALS OVER LAST 3 OBSERV.

```

```

C PHI          = DISCRETE-TIME STATE TRANSITION MATRIX
C PHIT         = TRANSPOSE OF PHI
C DEL          = STATE NOISE COEFFICIENT MATRIX
C DELT         = TRANSPOSE OF DEL
C PI           = 3.141592654
C PKK          = ESTIMATION ERROR COV.MATRIX, P(K|K)
C PKKS         = SMOOTHED ERROR COVARIANCE MATRIX
C PKKM1        = PREDICTED EST.ERROR COV.MATRIX, P(K|K-1)
C PKKM1S       = PREDICTED ERR.COVS.MAT.FOR SMOOT.P(K+1|K)
C IPKKM1S      = INVERSE OF PKKM1S
C PSS          = ERROR COV.MATRIX FOR SMOOTHING, P(K|K)
C R            = MEASUREMENT NOISE COVARIANCE
C RANGE        = DISTANCE FM SENSOR TO PRIORI TARGET POS.
C RTOD         = RADIAN TO DEGREE CONVERSION FACTOR
C SPD          = ESTIMATED TARGET SPEED IN KNOTS
C TEMP         = TEMPORARY STORAGE MATRICES USED IN
C              MATRIX OPERATIONS
C VARE         = VARIANCE OF RESIDUALS PROCESS
C XDIFF        = DISTANCE IN X DIRECTION FROM SENSOR TO A
C              PRIORI TARGET POSITION
C XKK          = ESTIMATED TARGET STATE VECTOR, X(K|K)
C XKKS         = SMOOTHED TARGET STATE VECTOR
C XKKM1        = PREDICTED TARGET STATE VECTOR, X(K|K-1)
C XKKM1S       = PRED.TARGET STATE VEC.FOR SMOOT.X(K+1|K)
C XPOS         = ESTIMATED TARGET POSITION IN X DIRECTION
C XS          = SENSOR POSITION IN X DIRECTION
C XSS         = TARGET STATE VEC.FOR SMOOTHING, X(K|K)
C XT           = TRUE TARGET POSITION IN X DIRECTION
C YDIFF        = DIST.IN Y DIRECTION FROM SENSOR TO A
C              PRIORI TARGET POSITION
C YPOS         = ESTIMATED TARGET POSITION IN Y DIRECTION
C YS          = SENSOR POSITION IN Y DIRECTION
C YT           = TRUE TARGET POSITION IN Y DIRECTION
C ZX           = OBSERVED POSITION IN X DIRECTION
C ZY           = OBSERVED POSITION IN Y DIRECTION

```

```

C ***VARIABLE DECLARATIONS***

```

```

    CHARACTER*1 A,B

```

```

    REAL*4 XKK(4,1),XKKM1(4,1),LPKKM1(4,4),LXKKM1(4,1)
    REAL*4 H(2,4),HT(4,2),G(4,2),TEMP1(2,1),TEMP2(2,4)
    REAL*4 TEMP3(2,1),TEMP4(4,2),TEMP5(4,1),TEMP6(4,4)
    REAL*4 TEMP7(4,4),TEMP8(4,1),TEMP9(4,1),TEMP10(4,2)
    REAL*4 DEL(4,2),DELT(2,4),UK(2,1)
    REAL*4 PKK(4,4),PKKM1(4,4),Z(2,1),BRG
    REAL*4 LXKK(4,1),LPKK(4,4),XS(10),YS(10),DBRG(10)
    REAL*4 PHI(4,4),PHIT(4,4),IMAT(4,4),XT,YT
    REAL*4 GATE1,E(2,1),VARE(2,2),IVARE(2,2),RTOD,DTOR
    REAL*4 DT,DTF,XDIFF,YDIFF,RANGE,XS1,YS1,BRG1,BRKKM1
    REAL*4 DATE,HR,MN,LAT,LONG,TOTIM,TIME,TIMEM1,DATE1

```

```

REAL*4  OBSERR(300),FAC1,SIGTH2,SIGVT2,R(2,2),ETOTAL
REAL*4  X2,YS2,BRG2,ZX,ZY,M1,E1,E1M1,E1M2,TRKERR(300)
REAL*4  M2,E2,E2M1,E2M2,G11,G13,G21,G23,ZXM1,ZYM1
REAL*4  XKKS(4,1,300),PKKS(4,4,300),EAVG
REAL*4  XNNM1(4,1),XSS(4,1),XKKM1S(4,1),AK(4,4)
REAL*4  PNNM1(4,4),PSS(4,4),PKKM1S(4,4),IPKKM1S(4,4)
REAL*4  AKT(4,4),II(4,4),STRKERR(300),DTS(300)
REAL*4  TEMP1S(4,4),TEMP2S(4,1),TEMP3S(4,1)
REAL*4  TEMP4S(4,4),TEMP5S(4,4),TEMP6S(4,4)
REAL*4  THETA,SEN,BEN,SIGRA2,SIGBE2
INTEGER*2 NP
INTEGER*2 PCN

```

C OPEN OUTPUT DATA FILES

```

OPEN(UNIT=2,FILE='STORM1.DAT',STATUS='OLD')
OPEN(UNIT=3,FILE='OUTDATA.DAT',STATUS='NEW')
OPEN(UNIT=4,FILE='TRUDATA.DAT',STATUS='NEW')
OPEN(UNIT=5,FILE='FILDATA.DAT',STATUS='NEW')
OPEN(UNIT=6,FILE='SMDATA.DAT',STATUS='NEW')
OPEN(UNIT=7,FILE='ELLIPDAT.DAT',STATUS='NEW')
OPEN(UNIT=8,FILE='MATRIX.DAT',STATUS='NEW')
OPEN(UNIT=9,FILE='ERRDATA.DAT',STATUS='NEW')
OPEN(UNIT=10,FILE='ERRSDATA.DAT',STATUS='NEW')

```

C RADIAN/DEGREE CONVERSION FACTORS

```

RTOD=57.29577951
DTOR=0.01745293

```

C COMPUTE 4X4 IDENTITY MATRIX

```

DO 5 I=1,4
DO 5 J=1,4
IF (I.EQ.J) THEN
    IMAT(I,J)=1.0
ELSE
    IMAT(I,J)=0.0
ENDIF
5 CONTINUE

DO 6 I=1,2
DO 6 J=1,4
    H(I,J)=0.0
6 CONTINUE
H(1,1)=1.0
H(2,3)=1.0

```

C INITIALIZE TIME COUNTER

```

TOTTIM=0.0
TIMEM1=0.0
NP=0

```

```

C INITIALIZE COUNTER FOR MANEUVER GATE
    E1M1=0.0
    E1M2=0.0

C COMPUTE BEARING MEASUREMENT COVARIANCE
C BEARING ERROR STANDARD DEVIATION = 1 NM
    WRITE(*,*) 'FILTERING OBSERVED DATA WITH KALMAN FILTER'
    WRITE(*,*) '***=====***'
810    READ(2,1001,END=800) DATE,HR,MN,LAT,A, LONG,B, PCN

1001    FORMAT(F6.0,F2.0,F2.0,F3.0,A1,F4.0,A1,I1)

C FOR VARIANCE RELATED OBS.NOISE R MATRIX VALUES, FOR
C SATELLITE DERIVED TROPICAL CYCLONES RANGE AND BEARING
C DEVIATIONS RANGE DEV.GETTING FROM 1988 ANNUAL CYCLONE
C REPORT BEARING BEARING DEV. ARE UNIFORMLY DISTRIBUTED

    IF((PCN.EQ.1).OR.(PCN.EQ.2)) THEN
        SIGRA2=184.96
        SIGBE2=0.822467
    ELSEIF((PCN.EQ.3).OR.(PCN.EQ.4)) THEN
        SIGRA2=292.41
        SIGBE2=0.822467
    ELSEIF((PCN.EQ.5).OR.(PCN.EQ.6)) THEN
        SIGRA2=948.64
        SIGBE2=0.822467
    ENDIF
C IF RADAR USES FOR STORM TRACKING
C     SIGRA2=361
C     SIGBE2=0.822467

    R(1,1)=SIGBE2
    R(1,2)=0.0
    R(2,1)=0.0
    R(2,2)=SIGRA2

    NP=NP+1

    MN=MN/60.0
    LAT=LAT/10
    LONG=LONG/10
    TIME=HR+MN

    IF (NP.EQ.1) THEN
        DATE1=DATE
        TIMEM1=TIME
    ENDIF

    IF (DATE.NE.DATE1) THEN
        TIME=TIME+24
        DT=TIME-TIMEM1

```

```

      TIME=TIME-24
    ELSE
      DT=TIME-TIMEM1
    ENDIF

```

```

      DTF=DT*60.0
      DTS(NP)=DT
      TOTTIM=TOTTIM+DT

```

```

      CALL INIT(LONG, LAT, XKK, PKK)

```

C IN ORDER TO APPLY CORRECT FORCING FUNCTION WE NEED TO KNOW  
 C STORM DIREC.FOR EVERY WHERE, FOLLOWING ROUTINE IS TO FIND  
 C STORM DIRECTIONS

```

      IF(((XKK(1,1).GE.0).AND.(XKK(3,1).GE.0).AND.(XKKM1(1,1).GE.0).AND.
*      (XKKM1(3,1).GE.0)).OR.((XKK(1,1).LE.0).AND.(XKK(3,1).GE.0)
*      .AND.(XKKM1(1,1).LE.0).AND.(XKKM1(3,1).GE.0)))THEN
        IF((XKKM1(1,1).GE.XKK(1,1)).AND.(XKKM1(3,1).GE.XKK(3,1)))THEN
          SEN=DTOR*(XKKM1(3,1)-XKK(3,1))
          BEN=DTOR*(XKKM1(1,1)-XKK(1,1))
          THETA=ATAN(SEN/BEN)
          THETA=RTOD*THETA
        ELSEIF((XKKM1(1,1).GE.XKK(1,1)).AND.(XKKM1(3,1).LE.XKK(3,1)))THEN
          SEN=DTOR*(XKKM1(3,1)-XKK(3,1))
          BEN=DTOR*(XKKM1(1,1)-XKK(1,1))
          THETA=90-ATAN(SEN/BEN)
          THETA=RTOD*THETA
        ELSE
          SEN=DTOR*(XKKM1(3,1)-XKK(3,1))
          BEN=DTOR*(XKKM1(1,1)-XKK(1,1))
          THETA=270-ATAN(SEN/BEN)
          THETA=RTOD*THETA
        ENDIF
      ELSEIF((XKK(1,1).GE.0).AND.(XKK(3,1).GE.0).AND.(XKKM1(1,1).LE.0)
*      .AND.(XKKM1(3,1).GE.0))THEN
        IF(XKKM1(3,1).LE.XKK(3,1))THEN
          SEN=DTOR*(XKKM1(3,1)-XKK(3,1))
          BEN=DTOR*(XKKM1(1,1)-XKK(1,1))
          THETA=270-ATAN(SEN/BEN)
          THETA=RTOD*THETA
        ENDIF
      ELSE
        SEN=DTOR*(XKKM1(3,1)-XKK(3,1))
        BEN=DTOR*(XKKM1(1,1)-XKK(1,1))
        THETA=90-ATAN(SEN/BEN)

```

```

      THETA=RTOD*THETA
ENDIF

```

C DIFFERENT FORCING FUNC.FOR DIFF.DIRECTION GROUP OF STORM

```

      IF((THETA.GE.265).AND.(THETA.LE.285)) THEN
        UK(1,1)=(1/200)*37*(XKKM1(3,1)-XKK(3,1))
        UK(2,1)=(1/200)*37*(XKKM1(3,1)-XKK(3,1))
      ELSEIF((THETA.GE.285).AND.(THETA.LE.345)) THEN
        UK(1,1)=0.245
        UK(2,1)=0.245
      ELSEIF((THETA.GE.345).AND.(THETA.LE.015)) THEN
        UK(1,1)=(1/200)*43*(XKKM1(3,1)-XKK(3,1))
        UK(2,1)=(1/200)*43*(XKKM1(3,1)-XKK(3,1))
      ELSEIF((THETA.GE.015).AND.(THETA.LE.025)) THEN
        UK(1,1)=0.245
        UK(2,1)=0.245
      ELSEIF((THETA.GE.025).AND.(THETA.LE.055)) THEN
        UK(1,1)=(1/200)*38*(XKKM1(3,1)-XKK(3,1))
        UK(2,1)=(1/200)*38*(XKKM1(3,1)-XKK(3,1))
      ELSE
        UK(1,1)=0.245
        UK(2,1)=0.245

```

```

ENDIF

```

```

      CALL FINDPHI(PHI,DT)
      CALL FINDDEL(DEL,DT)

```

```

      Z(1,1)=LONG
      Z(2,1)=LAT
      ZX=LONG
      ZY=LAT

```

```

      IF(NP.EQ.1) THEN
C          WRITE(*,*) 'X(0|0,0):'
          DO 601 I=1,4
            LXKK(I,1)=XKK(I,1)
C          WRITE(3,*) '*****'
C          WRITE(3,*) (XKK(I,1),I=1,4)
601      CONTINUE

```

```

C          WRITE(3,*) 'P(0|0,0):'
          DO 602 I=1,4
            DO 602 J=1,4
C          LPKK(I,J)=PKK(I,J)
C          WRITE(3,401) PKK(I,J)
401      FORMAT(4F14.4)

```

602

CONTINUE

ENDIF

C PROJECT AHEAD STATE AND ERROR COVARIANCE ESTIMATES

```

C      X(K+1|K) = PHI * X(K|K) + DEL * UK
C      CALL MATMUL(PHI,XKK,4,4,1,TEMP8)
C      CALL MATMUL(DEL,UK,4,2,1,TEMP9)
C      CALL MATADD(TEMP8,TEMP9,4,1,1,XKKM1)

C      WRITE(*,*) 'X(',TIME,'|',TIMEM1,',0):'
C      DO 603 I=1,4
C      WRITE(3,*) (XKKM1(I,1),I=1,4)
C      WRITE(3,*) '*****'
C      LXKKM1(I,1)=XKKM1(I,1)
603    CONTINUE

```

```

C      P(K+1|K) = (PHI * P(K|K) * PHIT) + Q

```

```

C      CALL MATRAN(PHI,PHIT,4,4)
C      CALL MATMUL(PHI,PKK,4,4,4,TEMP6)
C      CALL MATMUL(TEMP6,PHIT,4,4,4,TEMP7)
C      CALL GETQ(DT,XKKM1,Q,1)
C      CALL MATADD(TEMP7,Q,4,4,1,PKKM1)
C      DO 408 I=1,4
C      DO 408 J=1,4
C      LPKKM1(I,J)=PKKM1(I,J)
408    CONTINUE

C      WRITE(*,*) 'P(',TIME,'|',TIMEM1,',0):'
C      DO 604 I=1,4
C      WRITE(3,402) (PKKM1(I,J),J=1,4)
402    FORMAT(4F14.4)
604    CONTINUE

```

204 CONTINUE

C COMPUTE OBSERVATION RESIDUAL

```

C      E=Z(K)-H*X(K|K-1)
C      CALL MATMUL(H,XKKM1,2,4,1,TEMP1)
C      CALL MATSUB(Z,TEMP1,2,1,E)

```

C COMPUTE VARIANCE OF RESIDUALS SEQUENCE

C AND ADAPTIVE GATE VALUE

```

C      VAR(E)=H*PKKM1*HT+R
C      CALL MATRAN(H,HT,2,4)
C      CALL MATMUL(H,PKKM1,2,4,4,TEMP2)

```



```

        CALL MATMUL(TEMP2,HT,2,4,2,TEMP3)
        CALL MATADD(TEMP3,R,2,2,1,VARE)
C      WRITE(3,*) 'VARIANCE OF RESIDUALS = ',VARE
C      GATE1=1.5*SQRT(VARE)

C COMPUTE KALMAN GAIN MATRIX
C      G=PKKM1*HT*(H*PKKM1*HT+R)**-1
        CALL MATRAN(H,HT,2,4)
        CALL MATMUL(PKKM1,HT,4,4,2,TEMP4)
        CALL MATINV(VARE,2,IVARE)
        CALL MATMUL(TEMP4,IVARE,4,2,2,G)

C      WRITE(3,*) 'PKKM1*HT ='
        DO 414 I=1,4
C      WRITE(3,*) TEMP4(I,1)
414    CONTINUE

C      WRITE(3,*) 'G ='
        DO 613 I=1,4
C      WRITE(3,*) G(I,1)
613    CONTINUE

C      IF (L.EQ.1) THEN
C          G11=G(1,1)
C          G13=G(3,1)
C      ELSE
C          G21=G(1,1)
C          G23=G(3,1)
C      ENDIF

C COMPUTE UPDATED ESTIMATE
C      X(K|K)=X(K|K-1)+G*E, WHERE E=Z(K)-H*X(K|K-1)
        CALL MATMUL(G,E,4,2,1,TEMP5)
        CALL MATADD(TEMP5,XKKM1,4,1,1,XKK)

C      WRITE(3,*) 'X(',TIME,'|',TIME,',',L,'):'
        DO 605 I=1,4
C      WRITE(3,*) XKK(I,1)
605    CONTINUE

C COMPUTE UPDATED ERROR COVARIANCE MATRIX
C      P(K|K)=(I - G*H)*P(K|K-1)
        CALL MATMUL(G,H,4,2,4,TEMP6)
        CALL MATSUB(IMAT,TEMP6,4,4,TEMP7)
        CALL MATMUL(TEMP7,PKKM1,4,4,4,PKK)

C      WRITE(3,*) 'P(',TIME,'|',TIME,',',L,'):'
        DO 606 I=1,4
C      WRITE(3,406) (PKK(I,J),J=1,4)
406    FORMAT(4F14.4)

```

606           CONTINUE

C THESE STATEMENTS ARE FOR THE SMOOTHING ALGORITHM

                  DO 620 I=1,4  
                  XKKS(I,1,NP)=XKK(I,1)  
620               CONTINUE

                  DO 630 I=1,4  
                  DO 630 J=1,4  
                  PKKS(I,J,NP)=PKK(I,J)  
630               CONTINUE

C COMPUTE TRUE TRACKING ERROR

          TRKERR(NP)=SQRT((LAT-XKK(1,1))\*\*2+(LONG-XKK(3,1))\*\*2)

C COMPUTE OBSERVATION ERROR

C           OBSERR(NP)=SQRT((LAT-ZX)\*\*2+(LONG-ZY)\*\*2)

C COMPUTE ERROR ELLIPSE DATA

C CALL ELLIP(XKK(1,1),XKK(3,1),PKK(1,1),PKK(3,3),PKK(1,3))

C COMPUTE ESTIMATED X-Y POSITION, COURSE, AND SPEED

          XPOS=XKK(1,1)  
          YPOS=XKK(3,1)  
          IF (XKK(2,1).EQ.0 .AND. XKK(4,1).EQ.0) THEN  
              HDG=0.0  
          ELSE  
              HDG=RTOD\*ATAN2(XKK(2,1),XKK(4,1))  
          ENDIF  
          IF (HDG.LT.0.0) HDG=HDG+360  
          SPD=60\*SQRT(XKK(2,1)\*\*2+XKK(4,1)\*\*2)  
C       WRITE(\*,\*) 'FILTERED DATA FOR DATA POINT',NP  
          WRITE(3,\*) 'FILTERED DATA FOR DATA POINT',NP  
C       WRITE(\*,\*) 'TIME       X POS    Y POS   HEADING   SPEED'  
          WRITE(3,\*) 'TIME       X POS    Y POS   HEADING   SPEED'  
C       WRITE(\*,\*) TOTTIM,XPOS,YPOS,HDG,SPD  
          WRITE(3,\*) TOTTIM,XPOS,YPOS,HDG,SPD  
          WRITE(4,\*) TOTTIM,ZX,ZY  
          WRITE(5,\*) TOTTIM,XPOS,YPOS,PKK(1,1)  
          WRITE(9,\*) NP,TRKERR(NP)  
1002       FORMAT(1X,5F10.3)  
1003       FORMAT(1X,F6.2,3X,F10.1,2X,F11.1,3X,F8.1,3X,F8.1)  
1004       FORMAT(1X,F6.2,3(F8.1,2X))

C COMPARE BEARING ERRORS TO MANEUVER DETECTION GATES

          IF ((ABS(M1).GT.(GATE1))) THEN  
              WRITE(\*,\*) '\*\*\* MANEUVER DETECTION \*\*\*'

```

C          WRITE(3,*) '*** MANEUVER DETECTION ***'

CALL REINIT(DT,ZX,ZY,ZXM1,ZYM1,LPKKM1,XKKM1,PKKM1)
      E1M1=0.0
      E1M2=0.0
      GOTO 204
ENDIF

TIMEM1=TIME
DATE1=DATE

ZXM1=ZX
ZYM1=ZY

GOTO 810

C THIS IS WHERE THE SMOOTHING ALGORITHM STARTS
C FIXED INTERVAL SMOOTHING
800  WRITE(*,*) 'SMOOTHING FILTERED DATA WITH A'
      WRITE(*,*) 'FIXED INTERVAL SMOOTHING ALGORITHM'
      WRITE(*,*) '***=====***'

      DO 1000 KK=1,NP-1
        K=NP-KK

        DT=DTS(K+1)

        TIME=TIMEM1-DT
        CALL FINDPHI(PHI,DT)

        DO 901 I=1,4
          XSS(I,1)=XKKS(I,1,K)
901      CONTINUE

        DO 902 I=1,4
          DO 902 J=1,4
            PSS(I,J)=PKKS(I,J,K)
902      CONTINUE

C CALCULATE THE PREDIC.STATE AND ERROR COVARIANCE MATRICES
C      X(K+1|K)=PHI*X(K|K)
          CALL MATMUL (PHI,XSS,4,4,1,XKKM1S)
C      P(K+1|K)=PHI*P(K|K)*PHIT+Q
          CALL MATRAN (PHI,PHIT,4,4)
          CALL MATMUL(PHI,PSS,4,4,4,TEMP6)
          CALL MATMUL(TEMP6,PHIT,4,4,4,TEMP7)
          CALL GETQ(DT,XKKM1S,Q,1)
          CALL MATADD(TEMP7,Q,4,4,1,PKKM1S)

```

```

C CALCULATE THE SMOOTHING FILTER GAIN MATRIX
C   AK=P(K|K)*PHIT*INV{P(K+1|K)}
      CALL MATINV (PKKM1S,4,IPKKM1S)
      CALL MATMUL (PKKM1S,IPKKM1S,4,4,4,II)
      CALL MATMUL (PSS,PHIT,4,4,4,TEMP1S)
      CALL MATMUL (TEMP1S,IPKKM1S,4,4,4,AK)

      DO 904 I=1,4
        XNNM1(I,1)=XKKS(I,1,K+1)
904      CONTINUE

C CALCULATE THE SMOOTHED STATE ESTIMATE
C   XKKS=X(K|K)+AK*(X(K+1|N)-X(K+1|K))
      CALL MATSUB (XNNM1,XKKM1S,4,1,TEMP2S)
      CALL MATMUL (AK,TEMP2S,4,4,1,TEMP3S)
      CALL MATADD (XSS,TEMP3S,4,1,K,XKKS)

      DO 906 I=1,4
        DO 906 J=1,4
          PNNM1(I,J)=PKKS(I,J,K+1)
906      CONTINUE

C CALCULATE THE SMOOTHED COVARIANCE MATRIX
C   PKKS=P(K|K)+AK*[P(K+1|N)-P(K+1|K)]*AKT
      CALL MATSUB (PNNM1,PKKM1S,4,4,TEMP4S)
      CALL MATRAN (AK,AKT,4,4)
      CALL MATMUL (AK,TEMP4S,4,4,4,TEMP5S)
      CALL MATMUL (TEMP5S,AKT,4,4,4,TEMP6S)
      CALL MATADD (PSS,TEMP6S,4,4,K,PKKS)

C COMPUTE ESTIMATED X-Y POSITION, COURSE, AND SPEED
      SXPOS=XKKS(1,1,K)
      SYPOS=XKKS(3,1,K)
      IF (XKKS(2,1,K).EQ.0 .AND. XKKS(4,1,K).EQ.0) THEN
        SHDG=0.0
      ELSE
        SHDG=RTOD*ATAN2(XKKS(2,1,K),XKKS(4,1,K))
      ENDIF
      IF (SHDG.LT.0.0) SHDG=SHDG+360
      SSPD=60*SQRT(XKKS(2,1,K)**2+XKKS(4,1,K)**2)
C   WRITE(*,*) 'SMOOTHED DATA FOR DATA POINT',K
      WRITE(3,*) 'SMOOTHED DATA FOR DATA POINT',K
C   WRITE(*,*) 'TIME      X POS    Y POS    HEADING    SPEED'
      WRITE(3,*) 'TIME      X POS    Y POS    HEADING    SPEED'
C   WRITE(*,*) TOTTIM,SXPOS,SYPOS,SHDG,SSPD
      WRITE(3,*) TOTTIM,SXPOS,SYPOS,SHDG,SSPD
1010  FORMAT(1X,5F10.3)
1020  FORMAT(1X,F6.2,3X,F10.1,2X,F11.1,3X,F8.1,3X,F8.1)
1030  FORMAT(1X,F6.2,3(F8.1,2X))

      TIMEM1=TIME

```

1000 CONTINUE

C CALCULATE THE SMOOTHED TRACKING ERROR

```
C      OPEN(UNIT=4,FILE='TRUDATA.DAT',STATUS='OLD')
      DO 1100 K=1,NP
        SXPOS=XKKS(1,1,K)
        SYPOS=XKKS(3,1,K)
C      READ(4,1001)DATE,HR,MN,LAT,A, LONG,B,PCN
        STRKERR(K)=SQRT((LAT-SXPOS)**2+(LONG-SYPOS)**2)
        WRITE(6,1120)K,SXPOS,SYPOS,PKKS(1,1,K)
        WRITE(10,*)K,STRKERR(K)
```

1100 CONTINUE

1110 FORMAT(I4,2F8.1)

1120 FORMAT(I4,3(F8.1,2X))

1130 FORMAT(I4,3F8.1)

```
CLOSE(UNIT=2)
CLOSE(UNIT=3)
CLOSE(UNIT=4)
CLOSE(UNIT=5)
CLOSE(UNIT=6)
CLOSE(UNIT=7)
CLOSE(UNIT=8)
CLOSE(UNIT=9)
CLOSE(UNIT=10)
```

```
WRITE(*,*) 'FIL.& SM.OUTPUT DATA IS LOCATED IN THE'
WRITE(*,*) 'DATA FILE OUTDATA.DAT. FOR GRAPHIC
WRITE(*,*) RESULTS, 'ENSURE OBSDATA.DAT,
WRITE(*,*) FILDATA.DAT, & SMDATA.DAT ARE'
WRITE(*,*) 'IN THE MATLAB SUB-DIR.AND RUN THE
WRITE(*,*) MATLAB'M-FILE STORM2.M'
STOP
END
```

C\*\*\*\*\*

C SUBROUTINES

C\*\*\*\*\*

SUBROUTINE FINDPHI(PHI,DT)

C \*\*\*\*\*

C COMPUTES THE VALUES OF THE PHI MATRIX

C \*\*\*\*\*

REAL\*4 PHI(4,4),DT

DO 1501 I=1,4

DO 1501 J=1,4

DO 1501 K=1,2

```

                                PHI(I,J)=0.0
1501    CONTINUE

C COMPUTE PHI MATRIX
      DO 1500 I=1,4
        PHI(I,I)=1.0
1500    CONTINUE
        PHI(1,2)=DT
        PHI(3,4)=DT

      RETURN

    END

      SUBROUTINE FINDDEL(DEL,DT)
C*****
C  COMPUTE THE VALUES OF THE DEL MATRIX
C*****
      REAL*4 DEL(4,2),DT
      DEL(1,1)=DT**2./2.
      DEL(1,2)=0
      DEL(2,1)=DT
      DEL(2,2)=0
      DEL(3,1)=0
      DEL(3,2)=DT**2./2.
      DEL(4,1)=0
      DEL(4,2)=DT

      RETURN

    END

      SUBROUTINE INIT(LONG,LAT,XKK,PKK)
C *****
C  THIS ROUTINE INITIALIZES THE STATE
C  AND ERROR COVARIANCE ESTIMATES
C *****
      REAL*4 XKK(4,1),PKK(4,4)
      REAL*4 LAT, LONG

C INITIAL STATE ESTIMATE
      XKK(3,1)=LAT
      XKK(2,1)=0.0
      XKK(1,1)=LONG
      XKK(4,1)=0.0

C INITIAL ERROR COVARIANCE ESTIMATE
      PKK(1,1)=1000000
      PKK(1,2)=0.0

```

```

    PKK(1,3)=0.0
    PKK(1,4)=0.0
    PKK(2,1)=0.0
    PKK(2,2)=1000000
    PKK(2,3)=0.0
    PKK(2,4)=0.0
    PKK(3,1)=0.0
    PKK(3,2)=0.0
    PKK(3,3)=1000000
    PKK(3,4)=0.0
    PKK(4,1)=0.0
    PKK(4,2)=0.0
    PKK(4,3)=0.0
    PKK(4,4)=1000000

```

```

    RETURN

```

```

    END

```

```

    SUBROUTINE GETQ(DT,XKKM1,Q,FLAG)

```

```

C*****

```

```

C    ROUTINE TO GET Q MATRIX

```

```

C*****

```

```

    REAL*4 DT,XKKM1(4,1),Q(4,4)
    REAL*4 QPR(2,2),DEL(4,2),DELT(2,4)
    REAL*4 SIGVT2,SIGTH2,VT

```

```

    INTEGER FLAG

```

```

    IF ((XKKM1(2,1).EQ.0).OR.(XKKM1(4,1).EQ.0)) THEN

```

```

        DO 100 I=1,4

```

```

            DO 100 J=1,4

```

```

100          Q(I,J)=0.0

```

```

            GOTO 200

```

```

        ENDIF

```

```

C    CALCULATE Q' MATRIX

```

```

    IF((THETA.GE.265).AND.(THETA.LE.285)) THEN

```

```

        SIGVT2=230

```

```

        SIGTH2=280

```

```

    ELSEIF((THETA.GE.285).AND.(THETA.LE.345)) THEN

```

```

        SIGVT2=100

```

```

        SIGTH2=100

```

```

    ELSEIF((THETA.GE.345).AND.(THETA.LE.015)) THEN

```

```

        SIGVT2=210

```

```

        SIGTH2=540

```

```

    ELSEIF((THETA.GE.015).AND.(THETA.LE.025)) THEN

```

```

        SIGVT2=100

```

```

        SIGTH2=100

```

```

    ELSEIF((THETA.GE.025).AND.(THETA.LE.055)) THEN

```

```

        SIGVT2=220

```

```

        SIGTH2=370
ELSE
        SIGVT2=100
        SIGTH2=100
ENDIF

VT=SQRT(XKKM1(2,1)**2+XKKM1(4,1)**2)

QPR(1,1)=((XKKM1(2,1)/VT)**2)*SIGVT2+((XKKM1(4,1)**2)*SIGTH2)
QPR(2,2)=((XKKM1(4,1)/VT)**2)*SIGVT2+((XKKM1(2,1)**2)*SIGTH2)
QPR(1,2)=((XKKM1(2,1))*(XKKM1(4,1))/(VT**2))*SIGVT2
*      -(XKKM1(2,1))*(XKKM1(4,1))*SIGTH2

QPR(2,1)=QPR(1,2)
IF (FLAG.EQ.0) THEN
        QPR(1,1)=2.50*QPR(1,1)
        QPR(2,2)=2.50*QPR(2,2)
ENDIF

CALL FINDDEL(DEL,DT)

C Q=DEL(K)*Q'(K)*DELT(K)
CALL MATRAN(DEL,DELT,4,2)
CALL MATMUL(DEL,QPR,4,2,2,TEMP10)
CALL MATMUL(TEMP10,DELT,4,2,4,Q)
CALL MATSCL(0.01,Q,4,4,Q)

200    RETURN

END

SUBROUTINE REINIT(DT,ZX,ZY,ZXM1,ZYM1,LPKKM1,XKKM1,PKKM1)
C*****
C      THIS ROUTINE RE-INITIALIZES THE STATE AND ERROR
C      COVARIANCE ESTIMATES
C*****
      REAL*4 DT,XKKM1(4,1),PKKM1(4,4)
      REAL*4 ZX,ZY,ZXM1,ZYM1,LPKKM1(4,4)

      XDIFFF=ZX-ZXM1
      YDIFFF=ZY-ZYM1

      XKKM1(1,1)=ZX
      XKKM1(2,1)=XDIFFF/DT
      XKKM1(3,1)=ZY
      XKKM1(4,1)=YDIFFF/DT

C      WRITE(3,*) 'REINITIALIZED STATES ARE: '
      DO 100 I=1,4
C          WRITE(3,*) XKKM1(I,1)

```



100

CONTINUE

```

PKKM1(1,1)=2.25*LPKKM1(1,1)
PKKM1(1,2)=0.0
PKKM1(1,3)=2.25*LPKKM1(1,3)
PKKM1(1,4)=0.0
PKKM1(2,1)=0.0
PKKM1(2,2)=0.1111
PKKM1(2,3)=0.0
PKKM1(2,4)=0.0
PKKM1(3,1)=2.25*LPKKM1(3,1)
PKKM1(3,2)=0.0
PKKM1(3,3)=2.25*LPKKM1(3,3)
PKKM1(3,4)=0.0
PKKM1(4,1)=0.0
PKKM1(4,2)=0.0
PKKM1(4,3)=0.0
PKKM1(4,4)=0.1111

```

RETURN

END

SUBROUTINE MP(XS1,YS1,XS2,YS2,BRG1,BRG2,ZX,ZY)

C \*\*\*\*\*

C THIS ROUTINE COMPUTES THE ESTIMATED

C X,Y POSITION OBTAINED FROM MEASUREMENTS

C \*\*\*\*\*

REAL\*4 ZX,ZY

REAL\*4 XS1,YS1,XS2,YS2,BRG1,BRG2

REAL\*4 NUMER,DENOM

C INITIAL STATE ESTIMATE

NUMER=(-YS2\*TAN(BRG2))+(YS1\*TAN(BRG1))+XS2-XS1

DENOM=TAN(BRG1)-TAN(BRG2)

ZY=NUMER/DENOM

ZX=(ZY-YS1)\*TAN(BRG1)+XS1

RETURN

END

SUBROUTINE ELLIP(XT,YT,P1,P3,P13)

C \*\*\*\*\*

C THIS SUBROUTINE COMPUTES ERROR ELLIPSE DATA

C FROM ERROR COVARIANCE DATA

C \*\*\*\*\*

C DIMENSIONS AND DECLARATIONS

```

REAL*4 XT,YT,XP(21),YP(21),A,B,THE1,SIG2X,SIG2Y
REAL*4 SX,SY,PT,CT,ST,P1,P13,P3

A=2*P13
B=P1-P3
THE1=0.5*ATAN2(A,B)
A=(P1+P3)/2
B=0.0
IF (P13.EQ.0.0) GOTO 10
B=P13/SIN(2.0*THE1)
10 SIG2X=ABS(A+B)
SIG2Y=ABS(A-B)
SX=SIG2X**0.5
SY=SIG2Y**0.5
PT=3.141592654/10
CT=COS(THE1)
ST=SIN(THE1)

DO 100 IE=1,21
    XP(IE)=SX*COS(PT*IE)*CT-SY*SIN(PT*IE)*ST+XT
    YP(IE)=SX*COS(PT*IE)*ST+SY*SIN(PT*IE)*CT+YT
    WRITE(7,*)XP(IE),CHAR(9),YP(IE)
100 CONTINUE

RETURN

END

SUBROUTINE MATMUL(A,B,L,M,N,C)
C *****
C THIS ROUTINE MULTIPLIES TWO MATRICES TOGETHER
C { C(L,N) = A(L,M) * B(M,N) }
C *****
C DIMENSIONS AND DECLARATIONS
REAL*4 A(L,M),B(M,N),C(L,N)

DO 10 I=1,L
DO 10 J=1,N
    C(I,J)=0.0
10 CONTINUE

DO 100 I= 1,L
DO 100 J= 1,N
DO 100 K= 1,M
    C(I,J) = C(I,J) + A(I,K)*B(K,J)
100 CONTINUE

RETURN

END

```

```

      SUBROUTINE MATRAN(A,B,N,M)
C *****
C      THIS ROUTINE TRANSPOSES A MATRIX
C      ( B(M,N) = A'(N,M) )
C *****
C      DIMENSIONS AND DECLARATIONS
      REAL*4 A(N,M), B(M,N)

      DO 100 I= 1,N
      DO 100 J= 1,M
        B(J,I) = A(I,J)
100    CONTINUE

      RETURN

      END

      SUBROUTINE MATSCL(Q,A,N,M,C)
C *****
C      THIS ROUTINE MULTIPLIES A MATRIX WITH A SCALAR
C      ( C(N,M) = Q * A(N,M) )
C *****
C      DIMENSIONS AND DECLARATIONS
      REAL*4 A(N,M), C(N,M), Q

      DO 100 I = 1,N
      DO 100 J = 1,M
        C(I,J) = Q*A(I,J)
100    CONTINUE

      RETURN

      END

      SUBROUTINE MATSUB(A,B,N,M,C)
C *****
C      THIS ROUTINE SUBTRACTS TWO MATRICES
C      ( C(N,M) = A(N,M) - B(N,M) )
C *****
C      DIMENSIONS AND DECLARATIONS
      REAL*4 A(N,M), B(N,M), C(N,M)

      DO 100 I = 1,N
      DO 100 J = 1,M
        C(I,J)=A(I,J)-B(I,J)
100    CONTINUE

      RETURN

```

END

```
      SUBROUTINE MATADD(A,B,N,M,L,C)
C*****
C      THIS ROUTINE ADDS TWO MATRICES
C      { C(N,M) = A(N,M) + B(N,M) }
C*****
C      DIMENSIONS AND DECLARATIONS
      REAL*4  A(N,M),B(N,M),C(N,M,L)
      DO 100 I = 1,N
      DO 100 J = 1,M
        C(I,J,L)=A(I,J)+B(I,J)
100    CONTINUE

      RETURN
      END
```

```
      SUBROUTINE MATINV (A,N,C)
C*****
C      THIS ROUTINE COMPUTES THE INVERSE OF
C      A MATRIX
C      C(N,N)=INV [A(N,N)]
C*****
C      DIMENSIONS AND DECLARATIONS
      REAL*4 A(N,N),C(N,N),D(20,20)
      DO 100 I = 1,N
      DO 100 J = 1,N
100    D(I,J)=A(I,J)

      DO 115 I=1,N
      DO 115 J=N+1,2*N
115    D(I,J)=0.0

      DO 120 I=1,N
      J=I+N
120    D(I,J)=1.0

      DO 240 K=1,N
      M=K+1
      IF (K.EQ.N) GOTO 180
      L=K
      DO 140 I=M,N
140    IF (ABS(D(I,K)).GT.ABS(D(L,K))) L=I
      IF (L.EQ.K) GOTO 180

      DO 160 J=K,2*N
      TEMP=D(K,J)
      D(K,J)=D(L,J)
160    D(L,J)=TEMP
```

```

180      DO 185 J=M,2*N
185      D(K,J)=D(K,J)/D(K,K)

          IF (K.EQ.1) GOTO 220
          M1=K-1
          DO 200 I=1,M1
            DO 200 J=M,2*N
200      D(I,J)=D(I,J)-D(I,K)*D(K,J)

          IF (K.EQ.N) GOTO 260

220      DO 240 I=M,N
          DO 240 J=M,2*N
240      D(I,J)=D(I,J)-D(I,K)*D(K,J)

260      DO 265 I=1,N
          DO 265 J=1,N
            K=J+N
265      C(I,J)=D(I,K)

      RETURN
      END

```

### LIST OF REFERENCES

1. Chan, J.C.L., "*Identification of the Steering Flow for Tropical Cyclone Motion*," *Monthly Weather Review*, 113, 1985.
2. Mutaf A., "*A Kalman Filter for Storm Tracking*," Master's thesis, Naval Postgraduate School, Monterey, California, 1989.
3. Gelp, Arthur, *Applied Optimal Estimation*. MIT Press, Cambridge, Massachusetts, 1974.
4. Spehn Stephen, "*Noise Adaptation and Correlated Maneuver Gating of an Extended Kalman Filter*," Master's thesis, Naval Postgraduate School, Monterey, California, 1990.
5. Joint Typhoon Warning Center, Guam, Mariana Islands, 1988 *Annual Tropical Cyclone Report*. Naval Oceanography Command Center/Joint Typhoon Warning Center, 1988.
6. "*A Global View of Tropical Cyclones*," Naval Research Marine Meteorology Program, 1985.

### BIBLIOGRAPHY

Stanley I. Grossman, *Multivariable Calculus, Linear Algebra, and Differential Equations*. Harcourt Brace Jovonovich, Inc., 1986.

Anthony D. Whalen, *Detection of Signals in Noise*. Academic Press, Inc., New Jersey, 1971.

Athanasios Papoulis, *Probability, Random Variables, and Stochastic Process*. McGraw-Hill, Inc., New York, 1984.

Charles W. Therrien, *Decision Estimation and Classification*. John Wiley and Sons, New York, 1989.

Donald E. Kirk, *Optimal Control Theory*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1970.